신용카드 사용자 연체 예측

딥러닝 신경망 (DNN, Deep Neural Network)을 이용한 분류 모델

빅데이터융합전공 202213212 박소현
빅데이터융합전공 202213419 이화연

1.주제 선정 배경

경제

팍팍한 살림살이에... 카드 연체율 9년 만에 최고

강신 기자
입력 2024-03-18 11:17 | 수정 2024-03-18 15:56

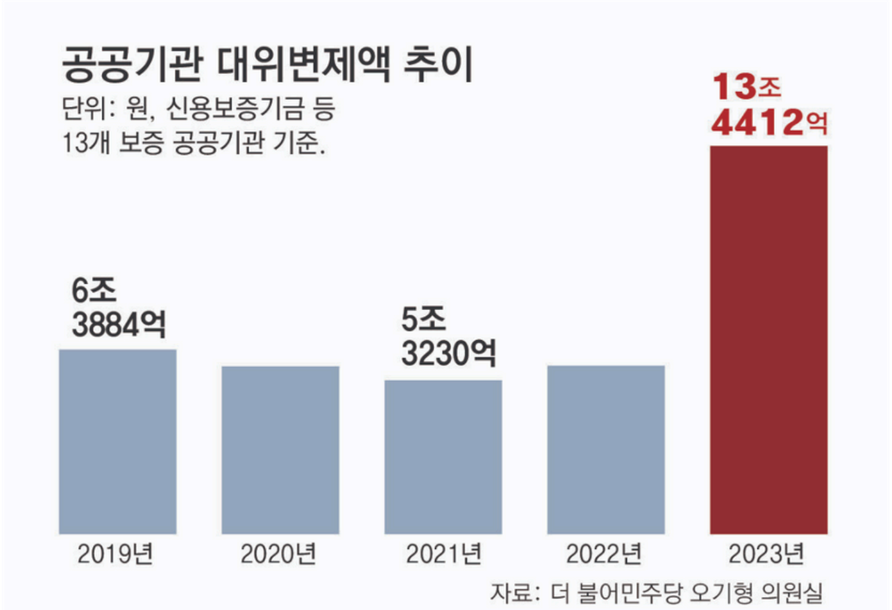
지난해 카드 연체율이 9년 만에 최고치로 뛰어올랐다.

금융감독원은 18일 '2023년 여신전문금융회사 영업실적(잠정)' 자료를 발표했다. 지난해 8개 전업카드사 연체율은 1.63%로 전년 말(1.21%)보다 0.42%포인트 상승해 2014년(1.69%) 이후 9년 만에 최고치를 기록했다. 그만큼 서민 생활이 팍팍해진 것으로 풀이된다.

카드사의 부실채권 비중도 급증했다. 카드사의 지난해 말 기준 고정이하여신비율은 1.14%로 전년 말보다 0.29%포인트 높아졌다.

강신 기자, "팍팍한 살림살이에... 카드 연체율 9년 만에 최고", 서울신문, 2024년 3월 18일

● 보증기관이 대신 갚은 빚만 13조 원



빚을 제때 못 갚는 서민이 늘어나면서 공공기관들이 은행 대신 빚을 갚아주는 경우도 크게 늘어났다. 국회 정무위원회 소속 더불어민주당 오기형 의원실에 따르면 신용보증기금, 주택도시보증공사, 지역신용보증재단 등 13개 보증 공공기관의 지난해 대위변제액은 13조4412억 원으로 2022년(5조8297억 원) 대비 130.6% 급증했다.

대위변제란 대출자가 원금을 상환하지 못했을 때 정책기관이 은행 대신 빚을 상환해주는 것을 말한다. 대위변제액은 2019년부터 2022년까지 연평균 5조8000억 원 수준을 유지하다가 지난해 폭발적으로 늘어났다.

강우석 기자, "카드 연체액 2조 넘어... 20년전 카드대란 육박", 동아일보, 2024년 5월 23일

2. 데이터 소개

raw data

	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
0	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	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	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	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
4	4	F	Y	Y	0	157500.0	State servant	Higher education	Married	House / apartment	-15037	-2105	1	0	0	0	Managers	2.0	-26.0	2.0
...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...
26452	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
26453	26453	F	N	Y	1	180000.0	Working	Higher education	Separated	House / apartment	-15291	-2475	1	0	0	0	NaN	2.0	-47.0	2.0
26454	26454	F	Y	N	0	292500.0	Working	Secondary / secondary special	Civil marriage	With parents	-10082	-2015	1	0	0	0	Core staff	2.0	-25.0	2.0
26455	26455	M	N	Y	0	171000.0	Working	Incomplete higher	Single / not married	House / apartment	-10145	-107	1	0	0	0	Laborers	1.0	-59.0	2.0
26456	26456	F	N	N	0	81000.0	Working	Secondary / secondary special	Civil marriage	House / apartment	-19569	-1013	1	0	0	0	Security staff	2.0	-9.0	2.0

26457 rows × 20 columns

2. 데이터 소개

변수 소개

이진 변수

gender: 성별

car: 차량 소유 여부

reality: 부동산 소유 여부

FLAG_MOBIL: 핸드폰 소유 여부

work_phone: 업무용 전화 소유 여부

phone: 전화 소유 여부

email: 이메일 소유 여부

gender의 고유값 : ['F' 'M']

car의 고유값 : ['N' 'Y']

reality의 고유값 : ['N' 'Y']

FLAG_MOBIL의 고유값 : [1]

work_phone의 고유값 : [0 1]

phone의 고유값 : [0 1]

email의 고유값 : [0 1]

2. 데이터 소개

변수 소개

연속형 변수

child_num: 자녀 수

income_total: 연간 소득

DAYS_BIRTH: 출생일

→ 데이터 수집 당시 (0)부터 역으로 셈. 즉, -1은 데이터 수집일 하루 전에 태어났음을 의미

DAYS_EMPLOYED: 업무 시작일

→ 데이터 수집 당시 (0)부터 역으로 셈. 즉, -1은 데이터 수집일 하루 전부터 일을 시작함을 의미 (양수 값은 고용되지 않은 상태)

begin_month: 신용카드 발급 월

→ 데이터 수집 당시 (0)부터 역으로 셈. 즉, -1은 데이터 수집일 한 달 전에 신용카드를 발급함을 의미

family_size: 가족 규모

2. 데이터 소개

변수 소개

다중 범주형 변수

income_type: 소득 분류

['Commercial associate', 'Working', 'State servant', 'Pensioner', 'Student']

edu_type: 교육 수준

['Higher education' ,'Secondary / secondary special', 'Incomplete higher', 'Lower secondary', 'Academic degree']

family_type: 결혼 여부

['Married', 'Civil marriage', 'Separated', 'Single / not married', 'Widow']

house_type: 생활 방식

['Municipal apartment', 'House / apartment', 'With parents', 'Co-op apartment', 'Rented apartment', 'Office apartment']

occyp_type: 직업 유형 (18개 종류)

2. 데이터 소개

변수 소개

다중 범주형 변수

income_type: 소득 분류

['Commercial associate', 'Working', 'State servant', 'Pensioner', 'Student']

- 상업적 종사자 / 일반 근로자 / 공무원 / 연금 수령자 / 학생

2. 데이터 소개

변수 소개

다중 범주형 변수

income_type: 소득 분류

['Commercial associate', 'Working', 'State servant', 'Pensioner', 'Student']

edu_type: 교육 수준

['Higher education' ,'Secondary / secondary special', 'Incomplete higher', 'Lower secondary', 'Academic degree']

- 대학교 졸업 / 고등학교 졸업 및 특수 교육 / 대학교 재학 중 / 중학교 졸업 / 석사, 박사 등 고급 학위 보유

2. 데이터 소개

변수 소개

다중 범주형 변수

income_type: 소득 분류

['Commercial associate', 'Working', 'State servant', 'Pensioner', 'Student']

edu_type: 교육 수준

['Higher education' ,'Secondary / secondary special', 'Incomplete higher', 'Lower secondary', 'Academic degree']

family_type: 결혼 여부

['Married', 'Civil marriage', 'Separated', 'Single / not married', 'Widow']

- 기혼 / 사실혼 / 별거 중 / 미혼 / 과부 또는 홀아비

2. 데이터 소개

변수 소개

다중 범주형 변수

income_type: 소득 분류

['Commercial associate', 'Working', 'State servant', 'Pensioner', 'Student']

edu_type: 교육 수준

['Higher education' ,'Secondary / secondary special', 'Incomplete higher', 'Lower secondary', 'Academic degree']

family_type: 결혼 여부

['Married', 'Civil marriage', 'Separated', 'Single / not married', 'Widow']

house_type: 생활 방식

['Municipal apartment', 'House / apartment', 'With parents', 'Co-op apartment', 'Rented apartment', 'Office apartment']

- 공공 임대 아파트 / 자가 주택 또는 아파트 / 부모와 함께 거주 / 협동 아파트 / 임대 아파트 / 사무실 아파트

2. 데이터 소개

변수 소개

종속변수

credit: 사용자의 신용카드 대금 연체를 기준으로 한 신용도
낮을수록 높은 신용의 신용카드 사용자를 의미

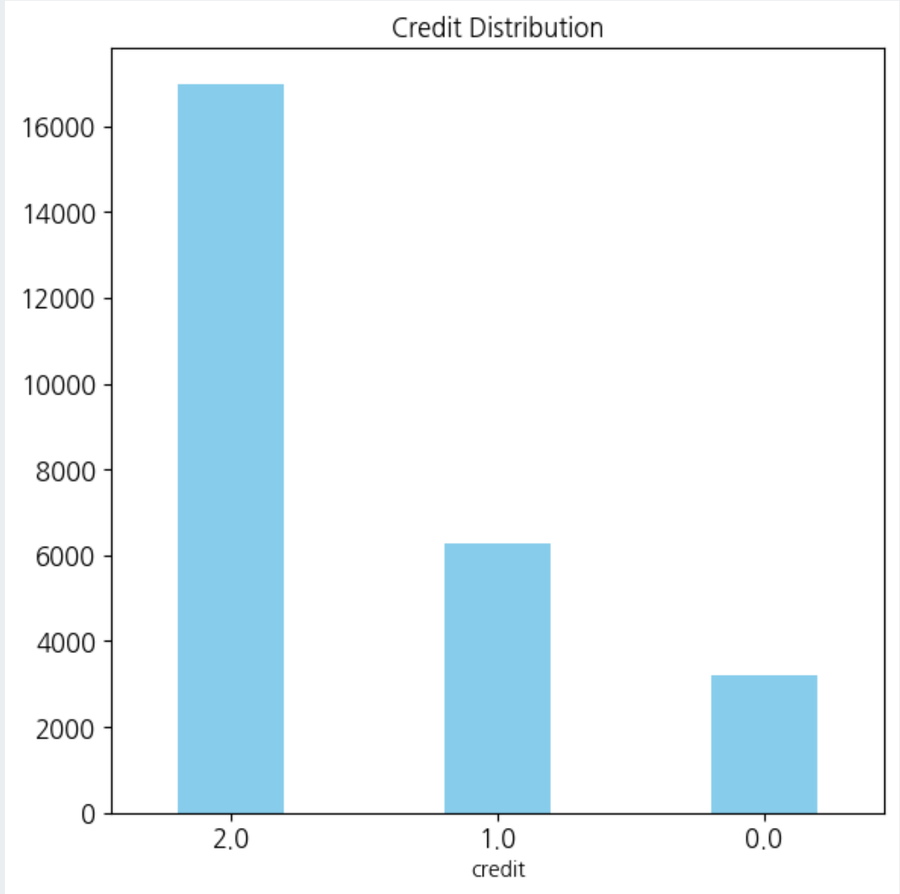
credit의 고유값 : [0. 1. 2.]

3. 데이터 전처리

.info()

```
Data columns (total 18 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  work_phone      26457 non-null  int64
 12  phone           26457 non-null  int64
 13  email           26457 non-null  int64
 14  occyp_type      18286 non-null  object
 15  family_size     26457 non-null  float64
 16  begin_month     26457 non-null  float64
 17  credit          26457 non-null  float64
dtypes: float64(4), int64(6), object(8)
```

credit



3. 데이터 전처리

DAYS_EMPLOYED

```
positive_count = (data['DAYS_EMPLOYED'] > 0).sum()
negative_count = (data['DAYS_EMPLOYED'] <= 0).sum()

print(f"양수 값 개수: {positive_count}")
print(f"음수 값 개수: {negative_count}")

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

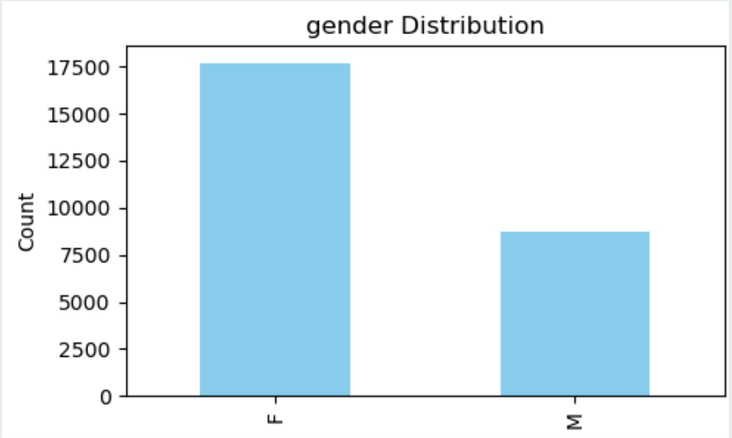
고용되지 않은 상태 : 4438
고용된 상태 : 22019

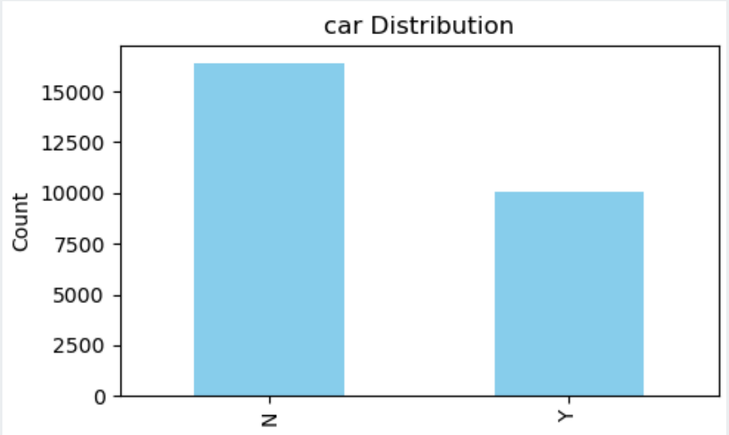
occyp_type 제거

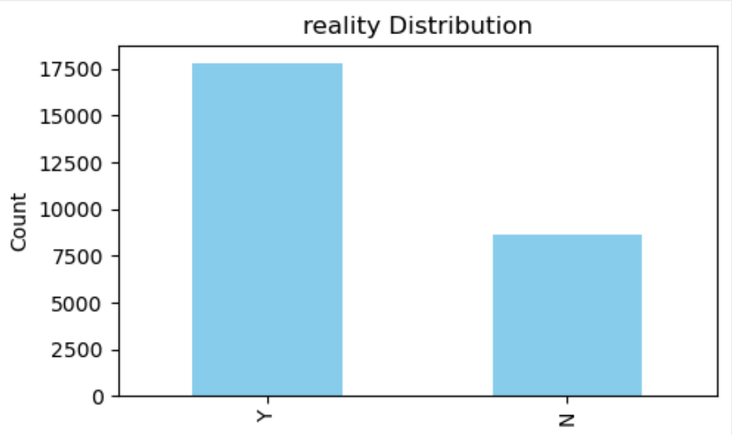
occyp_type	Accountants	Cleaning staff	Cooking staff	Core staff	Drivers	HR staff	High skill tech staff	IT staff	Laborers	Low-skill Laborers	Managers	Medicine staff	Private service staff	Realty agents	Sales staff	Secretaries	Security staff	Waiters/barmen staff
credit																		
0.0	5	2	3	15	8	0	5	0	26	1	12	4	2	0	13	0	2	0
1.0	5	2	3	14	8	0	6	0	25	0	11	4	1	0	15	1	2	1
2.0	5	2	2	14	9	0	6	0	24	1	12	5	1	0	14	0	3	1

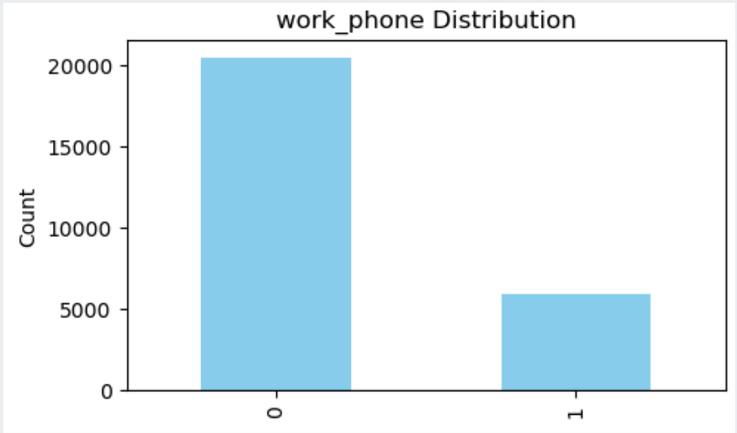
3. 데이터 전처리

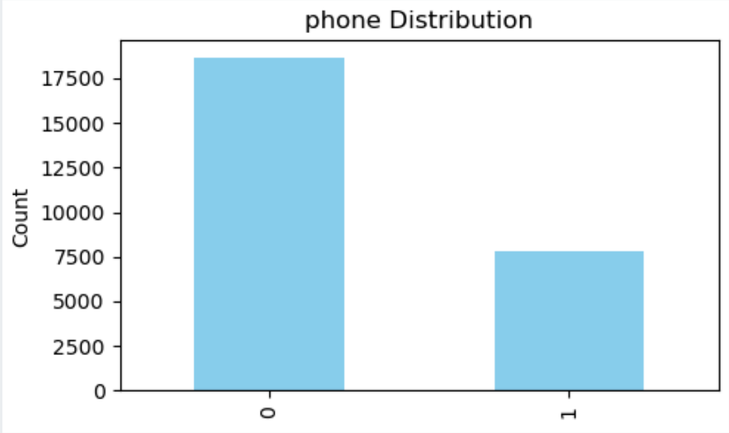
범주형 변수

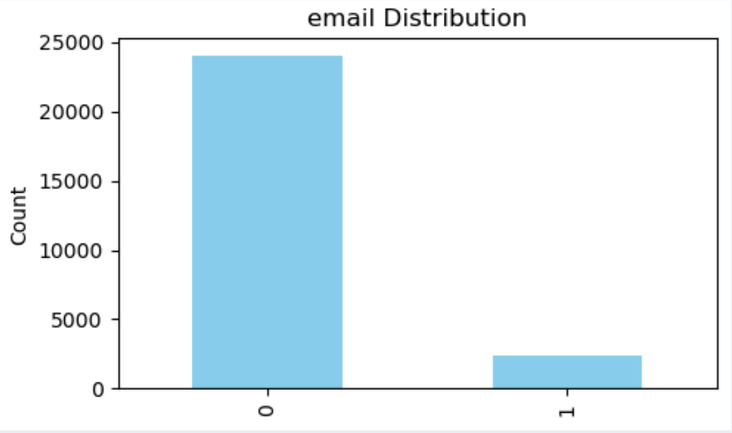






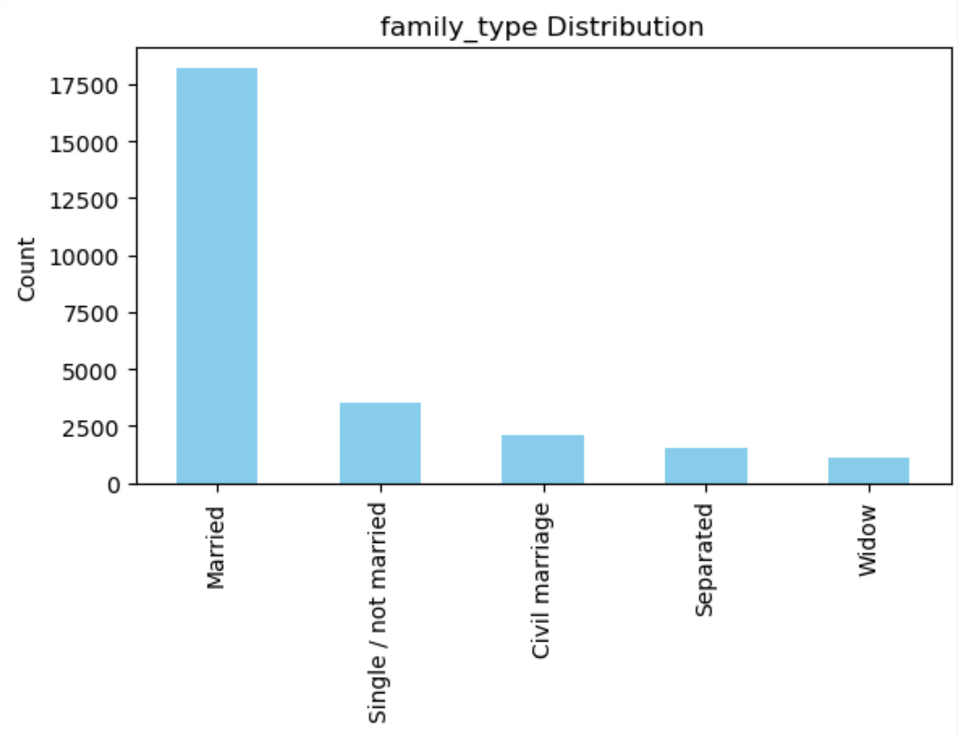


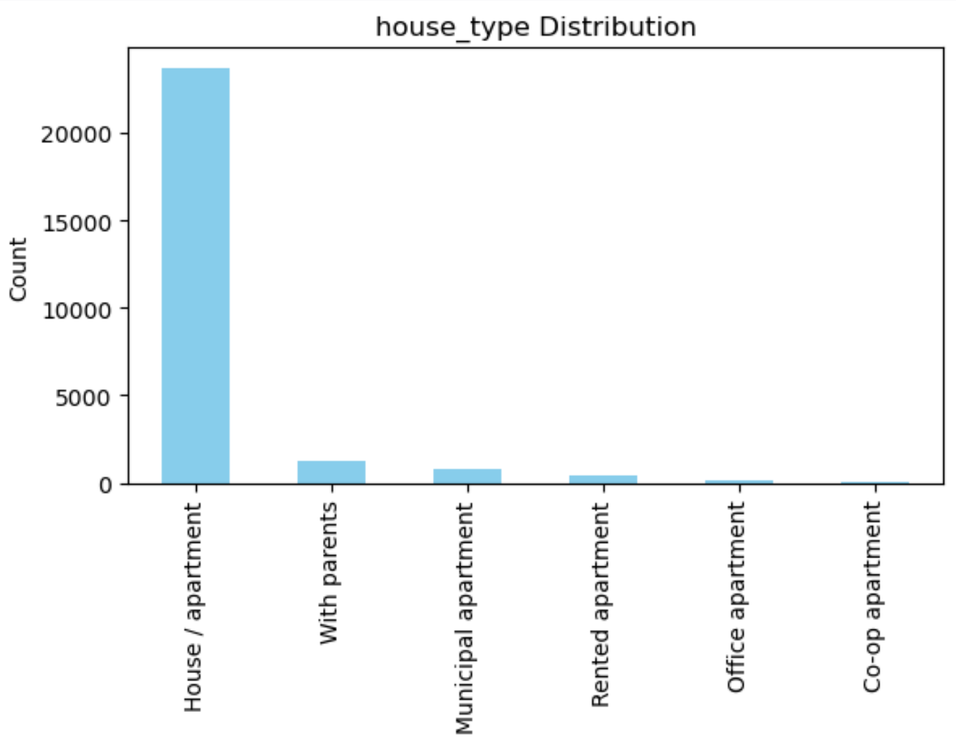




3. 데이터 전처리

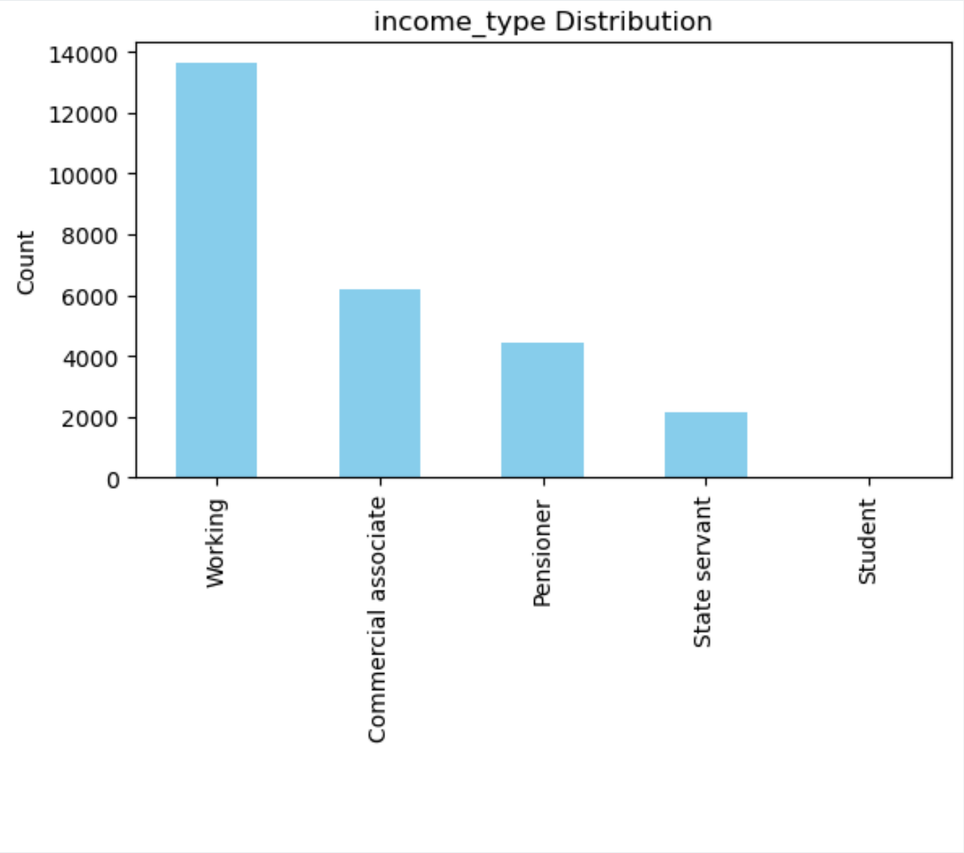
범주형 변수

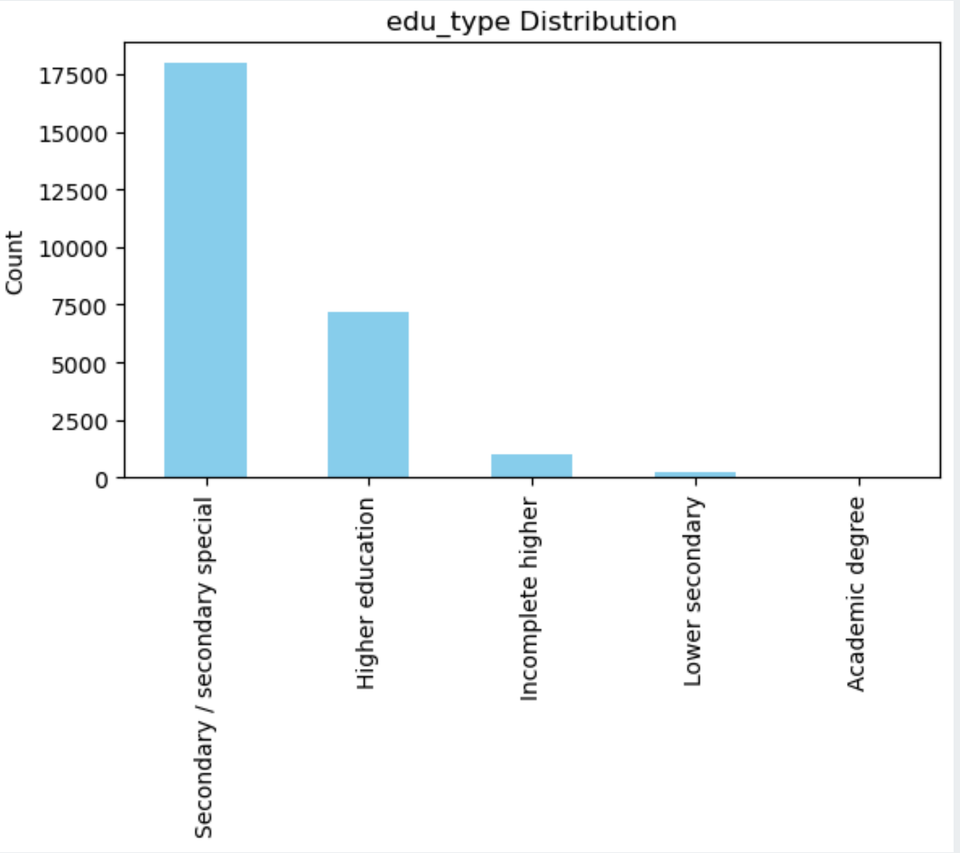




3. 데이터 전처리

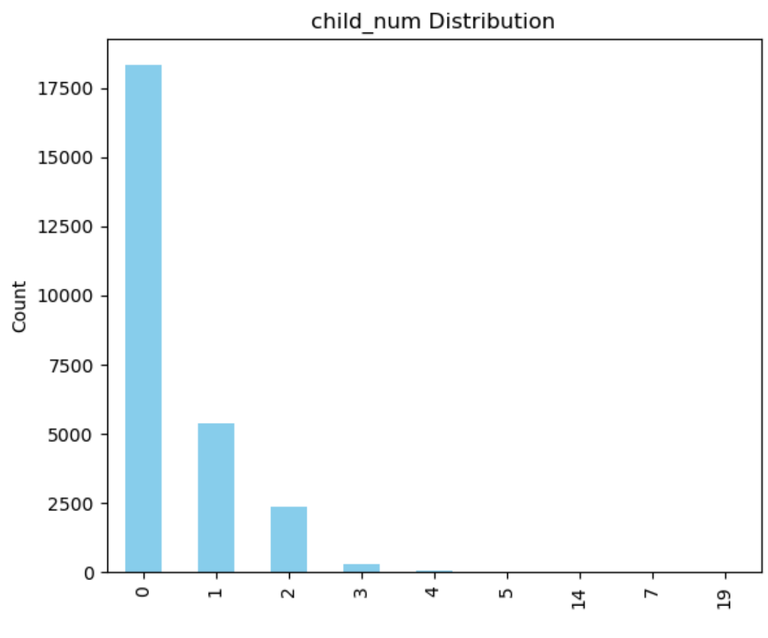
범주형 변수

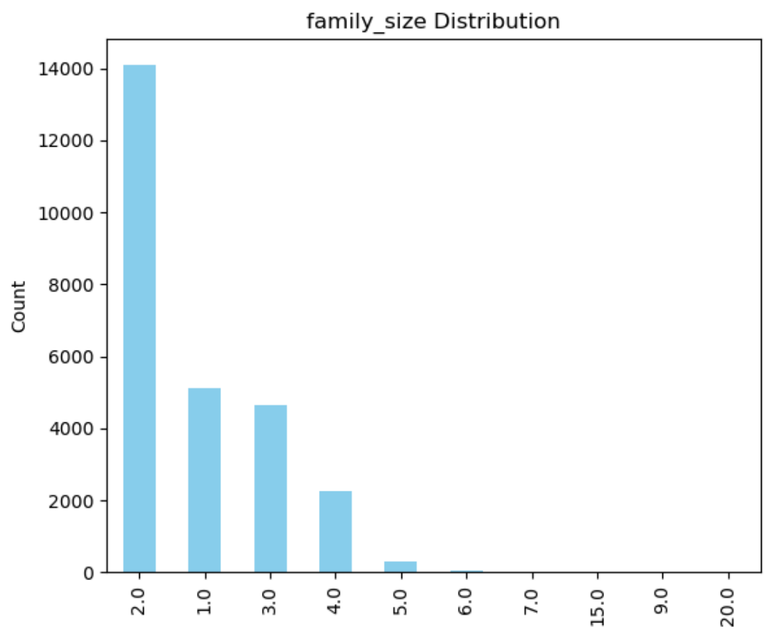




3. 데이터 전처리

family_size 제거





	child_num	family_size
child_num	1.00000	0.89053
family_size	0.89053	1.00000

```
data = data.drop('family_size', axis=1)
data['child_num'] = data['child_num'].apply(lambda x: 5 if x >= 5 else x)
```

3. 데이터 전처리

라벨인코딩

```
Data columns (total 18 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  work_phone      26457 non-null  int64
 12  phone           26457 non-null  int64
 13  email           26457 non-null  int64
 14  occyp_type      18286 non-null  object
 15  family_size     26457 non-null  float64
 16  begin_month     26457 non-null  float64
 17  credit          26457 non-null  float64
dtypes: float64(4), int64(6), object(8)
```

1. gender(F/M) int64로 바꾸기 -----> F : 0 , M : 1

2. car(N/Y) int64로 바꾸기 -----> N : 0 , Y : 1

3. reality(N/Y) int64로 바꾸기 -----> N : 0 , Y : 1

3. 데이터 전처리

절댓값 처리

```
Data columns (total 18 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  work_phone      26457 non-null  int64
 12  phone           26457 non-null  int64
 13  email           26457 non-null  int64
 14  occyp_type      18286 non-null  object
 15  family_size     26457 non-null  float64
 16  begin_month     26457 non-null  float64
 17  credit          26457 non-null  float64
dtypes: float64(4), int64(6), object(8)
```

4. 'DAYS_BIRTH', 'DAYS_EMPLOYED', 'begin_month'
   값을 절댓값(+)으로 바꾸기

3. 데이터 전처리

정수형 변환

```
Data columns (total 18 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  work_phone      26457 non-null  int64
 12  phone           26457 non-null  int64
 13  email           26457 non-null  int64
 14  occyp_type      18286 non-null  object
 15  family_size     26457 non-null  float64
 16  begin_month     26457 non-null  float64
 17  credit          26457 non-null  float64
dtypes: float64(4), int64(6), object(8)
```

5. 'float64 -> int64로 바꾸기

3. 데이터 전처리

라벨인코딩

```
Data columns (total 16 columns):
 #   Column          Non-Null Count  Dtype
---  ------          --------------  -----
 0   gender          26457 non-null  int64
 1   car             26457 non-null  int64
 2   reality         26457 non-null  int64
 3   child_num       26457 non-null  int64
 4   income_total    26457 non-null  int64
 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  work_phone      26457 non-null  int64
 12  phone           26457 non-null  int64
 13  email           26457 non-null  int64
 14  begin_month     26457 non-null  int64
 15  credit          26457 non-null  int64
dtypes: int64(12), object(4)
```

'income_type':

{'Commercial associate' : 0, 'Pensioner' : 1, 'State servant' : 2,
'Student' : 3, 'Working' : 4}

'edu_type':

{'Academic degree' : 0, 'Higher education' : 1, 'Incomplete higher' : 2,
'Lower secondary' : 3, 'Secondary / secondary special' : 4}

'family_type':

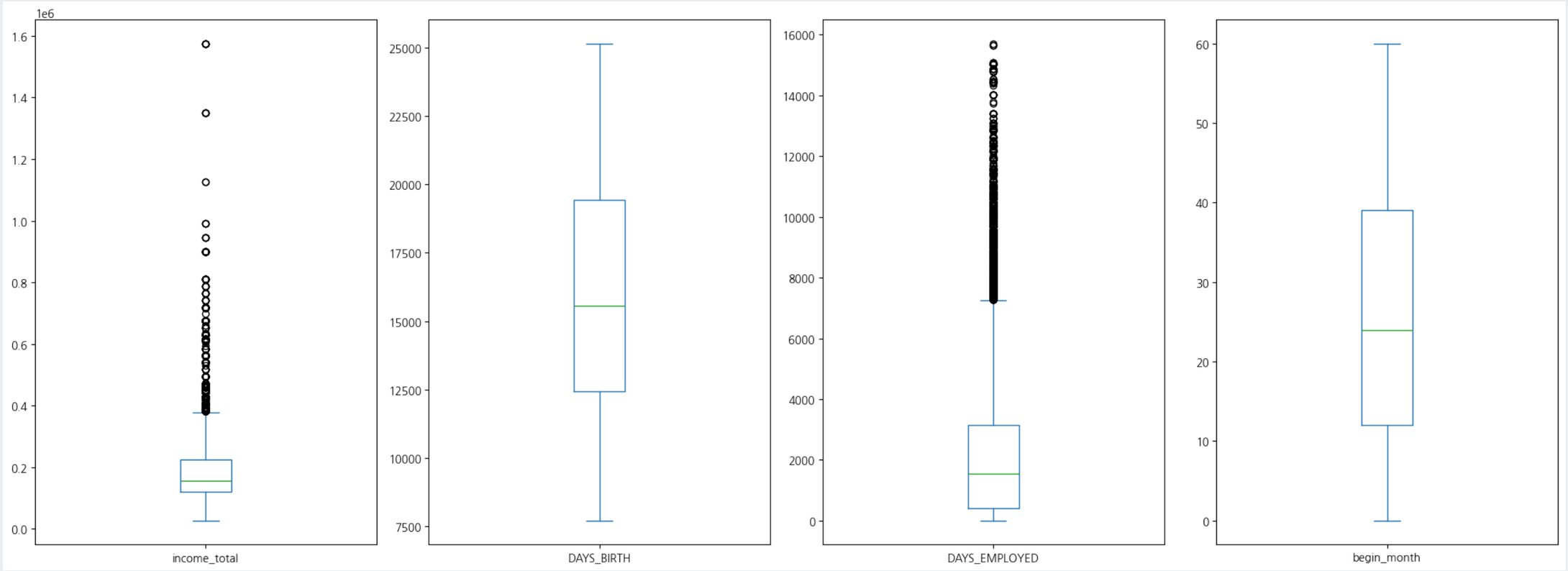
{'Civil marriage' : 0, 'Married' : 1, 'Separated' : 2,
'Single / not married' : 3, 'Widow' : 4}

'house_type':

{'Co-op apartment' : 0, 'House / apartment' : 1, 'Municipal apartment' : 2,
'Office apartment' : 3, 'Rented apartment' : 4, 'With parents' : 5}

3. 데이터 전처리

이상치 확인



3. 데이터 전처리

final data

전체 변수가 포함된 데이터프레임

	gender	car	reality	child_num	income_total	income_type	edu_type	family_type	house_type	DAYS_BIRTH	DAYS_EMPLOYED	work_phone	phone	email	begin_month	credit
0	0	0	0	0	202500	0	1	1	2	13899	4709	0	0	0	6	1
1	0	0	1	1	247500	0	4	0	1	11380	1540	0	0	1	5	1
2	1	1	1	0	450000	4	1	1	1	19087	4434	0	1	0	22	2
3	0	0	1	0	202500	0	4	1	1	15088	2092	0	1	0	37	0
4	0	1	1	0	157500	2	1	1	1	15037	2105	0	0	0	26	2
...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...
26452	0	0	0	2	225000	2	4	1	1	12079	1984	0	0	0	2	1
26453	0	0	1	1	180000	4	1	2	1	15291	2475	0	0	0	47	2
26454	0	1	0	0	292500	4	4	0	5	10082	2015	0	0	0	25	2
26455	1	0	1	0	171000	4	2	3	1	10145	107	0	0	0	59	2
26456	0	0	0	0	81000	4	4	0	1	19569	1013	0	0	0	9	2

26457 rows × 16 columns

3. 데이터 전처리

카이제곱 검정

```
# 이진 범주형 변수와 다중 범주형 변수 목록
binary_vars = ['gender', 'car', 'reality', 'work_phone', 'phone', 'email']
multi_category_vars = ['income_type', 'edu_type', 'family_type', 'house_type']
continuous_vars = ['child_num', 'income_total', 'DAYS_BIRTH', 'DAYS_EMPLOYED', 'begin_month']

# 다중 범주형 변수에 대해 원-핫 인코딩 수행
data_encoded = pd.get_dummies(data, columns=multi_category_vars)

# 결과를 저장할 리스트 생성
chi2_results = []

# 이진 변수에 대해 카이제곱 검정 수행
for var in binary_vars:
    contingency_table = pd.crosstab(data_encoded[var], data_encoded['credit'])
    chi2_stat, p_val, dof, _ = chi2_contingency(contingency_table)
    if p_val <= 0.05:  # 유의미한 변수만 저장
        chi2_results.append({
            'Variable': var,
            'Chi2 Statistic': chi2_stat,
            'p-value': round(p_val, 4),
            'Degrees of Freedom': dof
        })
```

```
# 다중 범주형 변수에 대해 원-핫 인코딩된 각 범주별로 카이제곱 검정 수행
for var in multi_category_vars:
    encoded_cols = [col for col in data_encoded.columns if col.startswith(var)]
    for col in encoded_cols:
        contingency_table = pd.crosstab(data_encoded[col], data_encoded['credit'])
        chi2_stat, p_val, dof, _ = chi2_contingency(contingency_table)
        if p_val <= 0.05:  # 유의미한 변수만 저장
            chi2_results.append({
                'Variable': col,
                'Chi2 Statistic': chi2_stat,
                'p-value': round(p_val, 4),
                'Degrees of Freedom': dof
            })

# 유의미한 변수들에 대한 카이제곱 검정 결과를 데이터프레임으로 변환
chi2_results_df = pd.DataFrame(chi2_results)

# 최종 데이터프레임 생성: 유의미한 변수와 연속형 변수 결합
final_vars = chi2_results_df['Variable'].tolist() + continuous_vars
final_data = data_encoded[final_vars]
```

3. 데이터 전처리

카이제곱 검정

	Variable	Chi2 Statistic	p-value
0	car	9.396436	0.0091
1	reality	11.231612	0.0036
2	phone	8.035167	0.0180
3	email	6.065864	0.0482
4	income_type_0	18.326786	0.0001
5	income_type_4	11.070212	0.0039
6	edu_type_1	7.017194	0.0299
7	family_type_0	9.901679	0.0071
8	family_type_1	29.757092	0.0000
9	family_type_3	22.936887	0.0000
10	family_type_4	7.498292	0.0235
11	house_type_2	8.262940	0.0161
12	house_type_4	27.572270	0.0000

이진 - 선택 O

차량 소유 여부

부동산 소유 여부

전화 소유 여부

이메일 소유 여부

이진 - 선택 X

성별

업무용 전화 소유 여부

다중 - 선택 O

라벨인코딩 → 원핫인코딩이므로 23페이지 참조

소득 분류 - 상업적 종사자 / 일반 근로자

교육 수준 - 대학교 졸업

결혼 여부 - 기혼 / 사실혼 / 미혼 / 과부 또는 홀아비

생활 방식 - 공공 임대 아파트 / 임대 아파트

3. 데이터 전처리

final data

중요한 변수만 포함된 데이터프레임

	car	reality	phone	email	income_type_0	income_type_4	edu_type_1	family_type_0	family_type_1	family_type_3	family_type_4	house_type_2	house_type_4	child_num	income_total
0	0	0	0	0	True	False	True	False	True	False	False	True	False	0	202500
1	0	1	0	1	True	False	False	True	False	False	False	False	False	1	247500
2	1	1	1	0	False	True	True	False	True	False	False	False	False	0	450000
3	0	1	1	0	True	False	False	False	True	False	False	False	False	0	202500
4	1	1	0	0	False	False	True	False	True	False	False	False	False	0	157500
...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...
26452	0	0	0	0	False	False	False	False	True	False	False	False	False	2	225000
26453	0	1	0	0	False	True	True	False	False	False	False	False	False	1	180000
26454	1	0	0	0	False	True	False	True	False	False	False	False	False	0	292500
26455	0	1	0	0	False	True	False	False	False	True	False	False	False	0	171000
26456	0	0	0	0	False	True	False	True	False	False	False	False	False	0	81000

26457 rows × 19 columns

4. 모델 학습 및 분석 결과

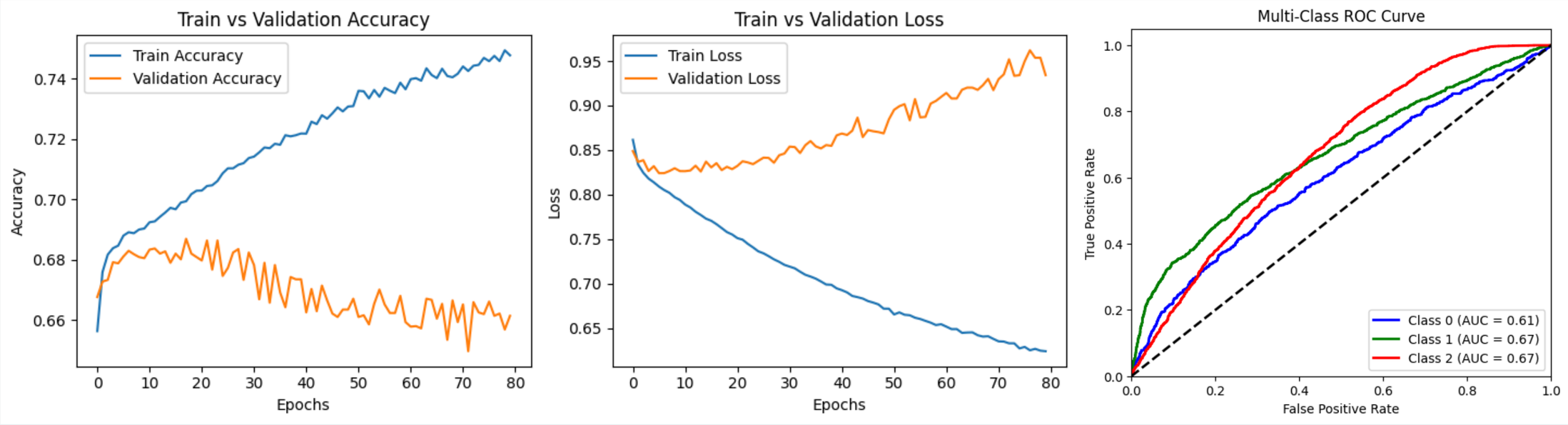
DNN 모델 학습

```
# 데이터 분할
X_train, X_test, y_train, y_test = train_test_split(X, y_encoded, test_size=0.2, random_state=42)

# 스케일러 학습 및 변환
columns_to_scale = ['DAYS_BIRTH', 'DAYS_EMPLOYED', 'income_total', 'begin_month', 'child_num']
scaler = StandardScaler()
X_train[columns_to_scale] = scaler.fit_transform(X_train[columns_to_scale])
X_test[columns_to_scale] = scaler.transform(X_test[columns_to_scale])

# DNN 모델 생성
dnn_model2 = Sequential([Dense(64, input_dim=X.shape[1], activation='relu'),
                         Dense(32, activation='relu'),
                         Dense(32, activation='relu'),
                         Dense(16, activation='relu'),
                         Dense(3, activation='softmax')])# 3개의 클래스에 대한 확률 출력

dnn_model2.compile(optimizer=Nadam(learning_rate=0.001), loss='categorical_crossentropy', metrics=['accuracy'])

# EarlyStopping 콜백 설정
early_stopping = EarlyStopping(
    monitor='val_loss',  # 'val_loss'를 모니터링
    patience=10,         # 개선되지 않는 에포크 수 (10 에포크 동안 개선 없으면 종료)
    restore_best_weights=True)  # 최상의 가중치를 복원

# 모델 학습
history = dnn_model2.fit(
    X_train, y_train,
    epochs=80,
    batch_size=32,
    validation_data=(X_test, y_test),
    callbacks=[early_stopping])  # 얼리스탑핑 콜백 추가
```

4. 모델 학습 및 분석 결과

DNN 분석 결과

변수선택(원핫인코딩)



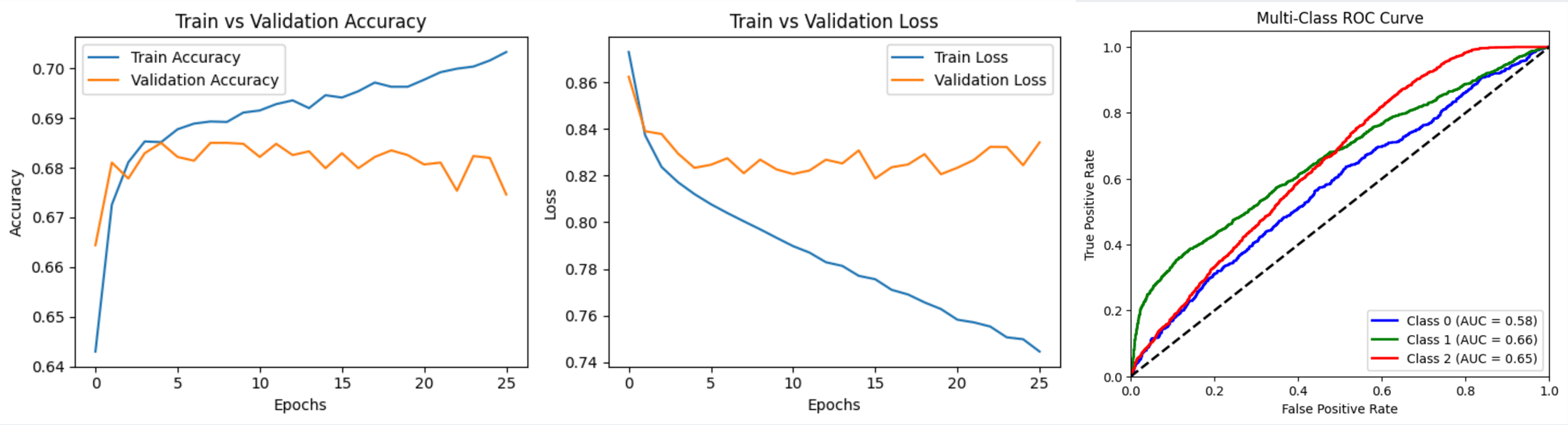
훈련 데이터 정확도 : 0.7573
검증 데이터 정확도 : 0.6614

F1 Score : 0.6140
Overall AUC Score : 0.6492

4. 모델 학습 및 분석 결과

DNN 분석 결과

변수선택 + EarlyStopping



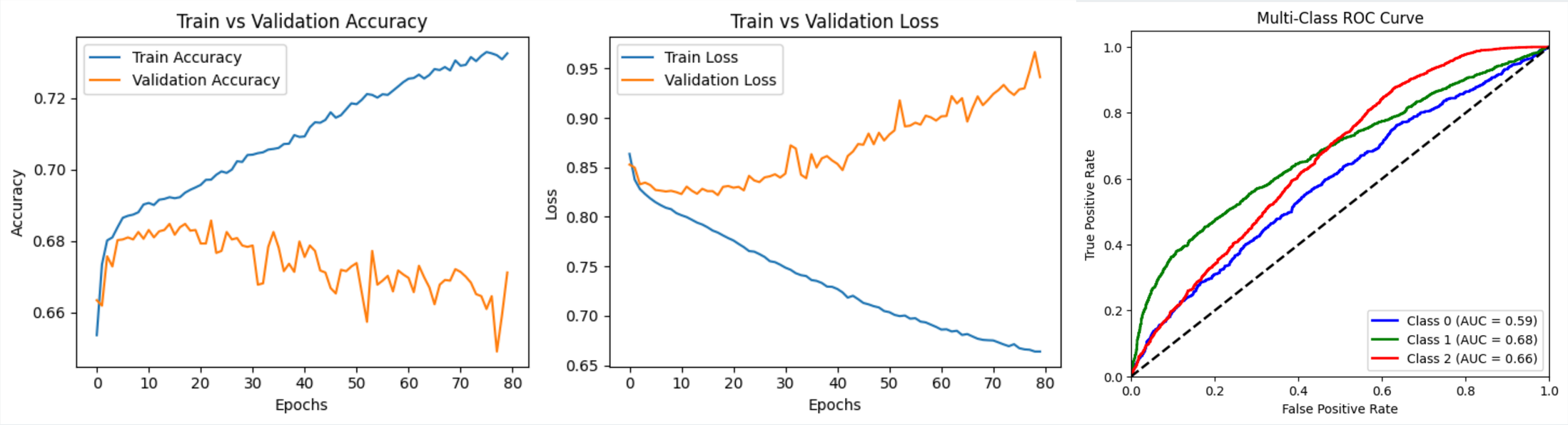
훈련 데이터 정확도 : 0.6977
검증 데이터 정확도 : 0.6829

F1 Score : 0.5956
Overall AUC Score : 0.6292

4. 모델 학습 및 분석 결과

DNN 분석 결과

변수전체(라벨인코딩)



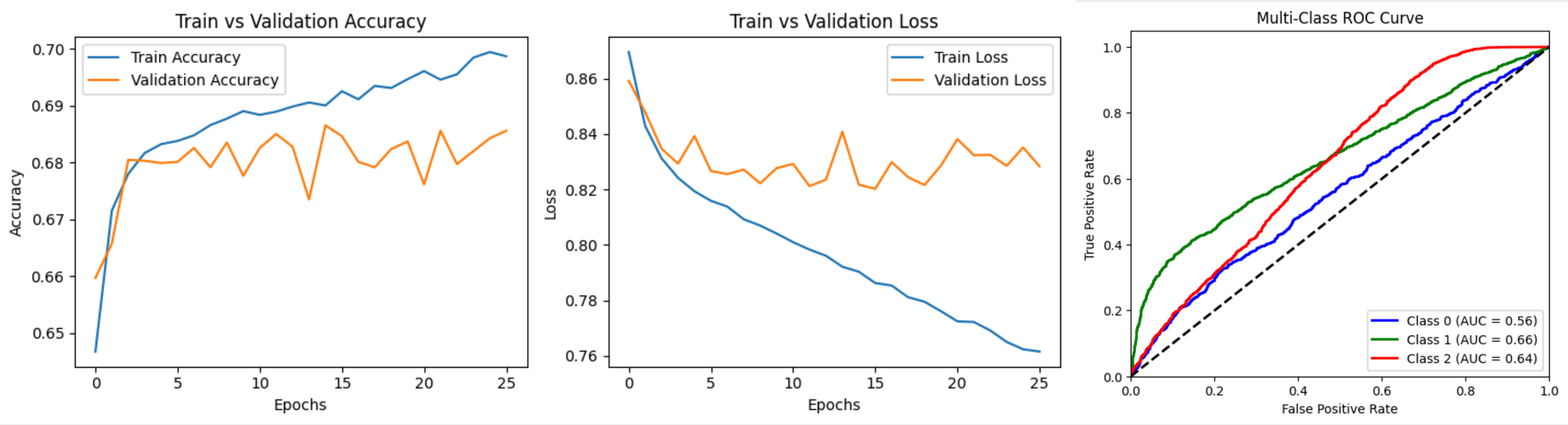
훈련 데이터 정확도 : 0.7417
검증 데이터 정확도 : 0.6712

F1 Score : 0.6137
Overall AUC Score : 0.6436

4. 모델 학습 및 분석 결과

DNN 분석 결과

변수전체 + EarlyStopping



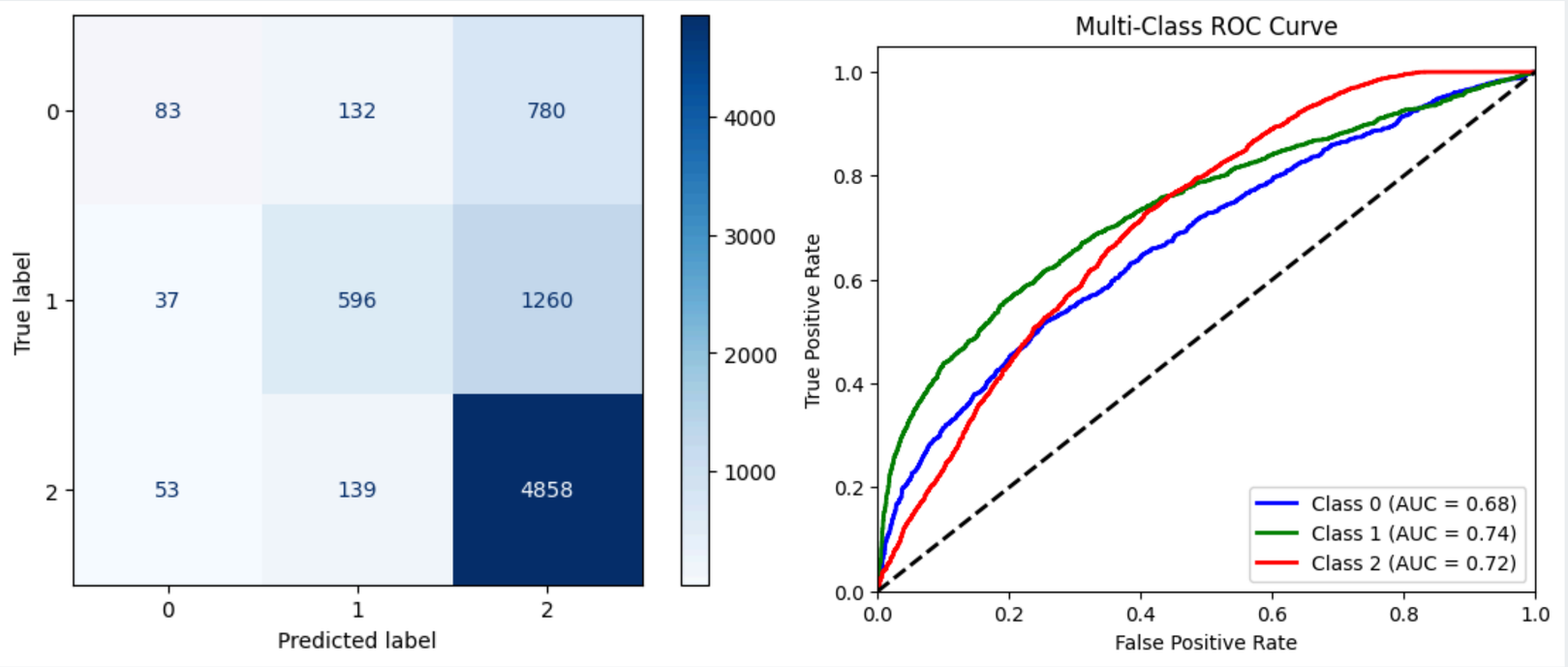
훈련 데이터 정확도 : 0.6940
검증 데이터 정확도 : 0.6846

F1 Score : 0.5997
Overall AUC Score : 0.6216

4. 모델 학습 및 분석 결과

XGBoost

변수선택(원핫인코딩)

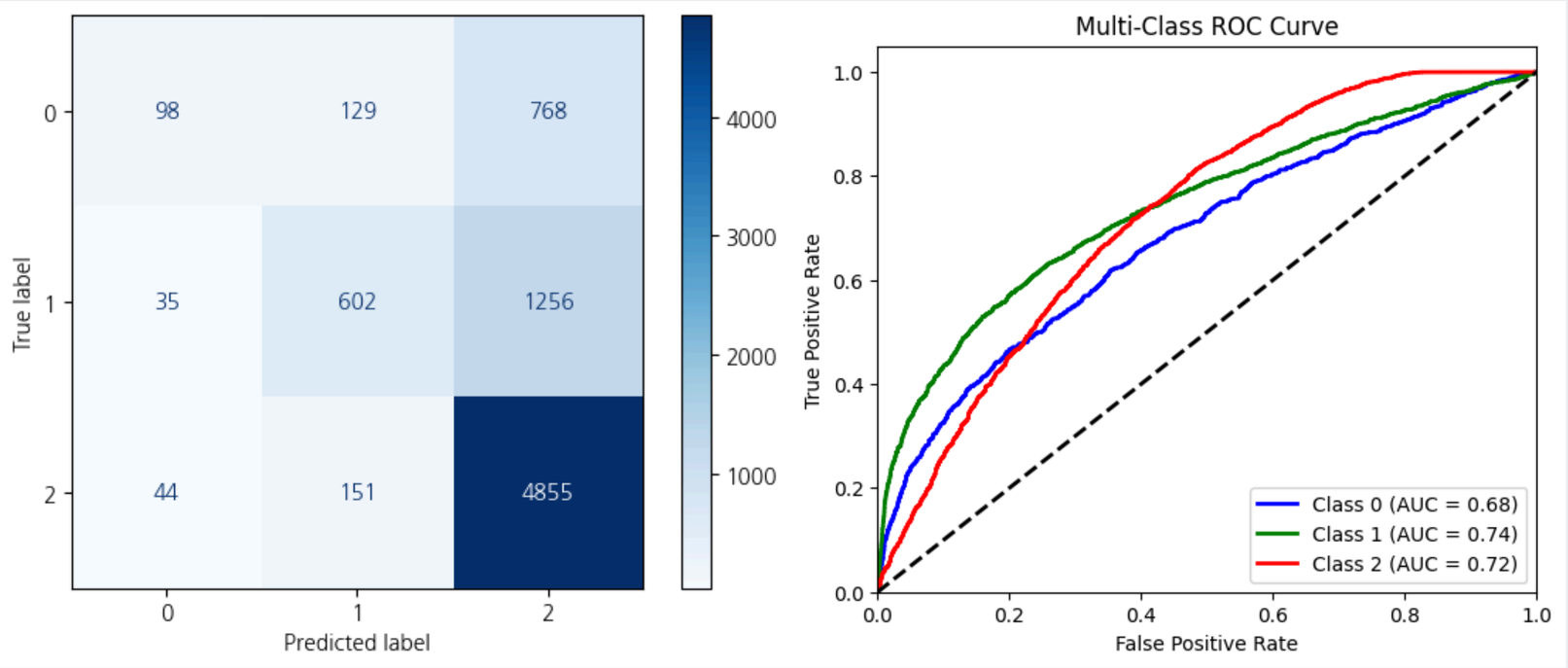


훈련 데이터 정확도 : 0.7774
검증 데이터 정확도 : 0.6975
F1 Score : 0.6381
AUC Score : 0.7094

4. 모델 학습 및 분석 결과

XGBoost

변수전체(라벨인코딩)

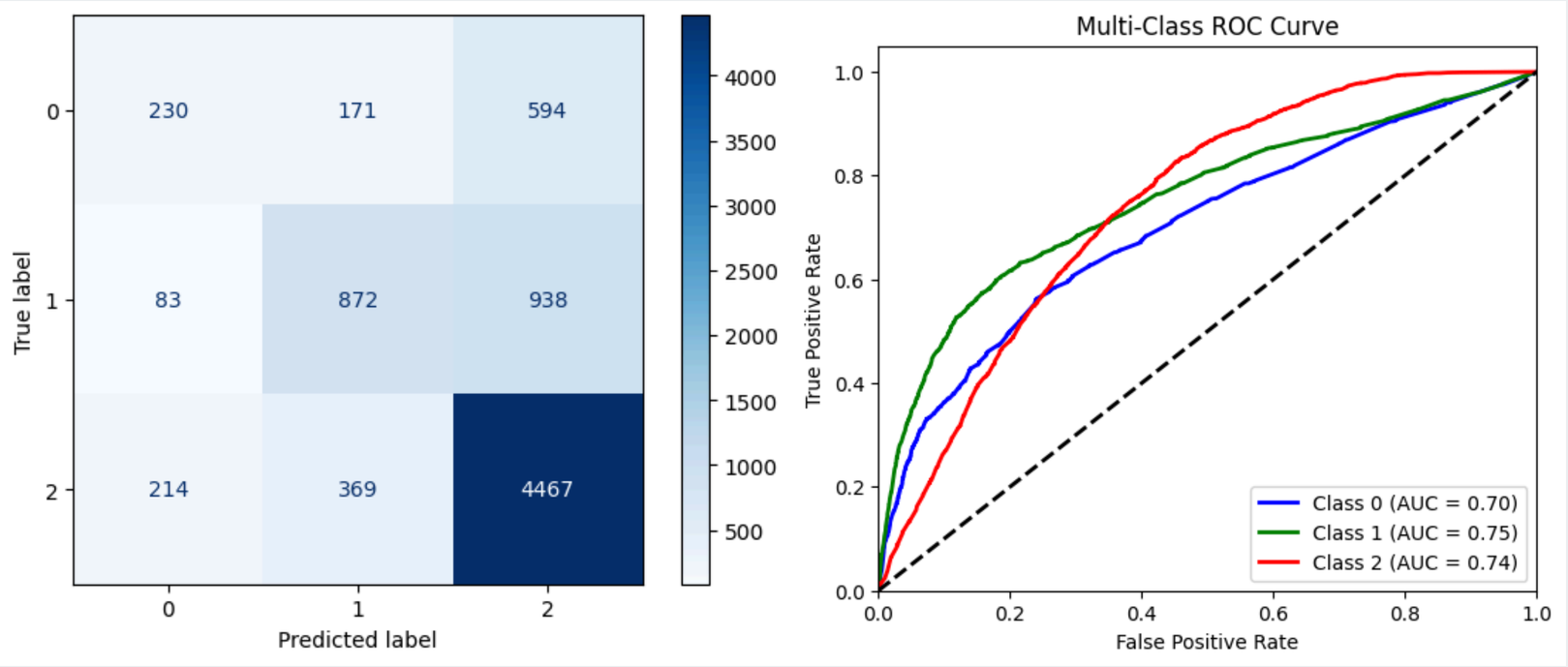


훈련 데이터 정확도 : 0.7841
검증 데이터 정확도 : 0.6998
F1 Score : 0.6423
AUC Score : 0.7151

4. 모델 학습 및 분석 결과

RandomForest

변수선택(원핫인코딩)

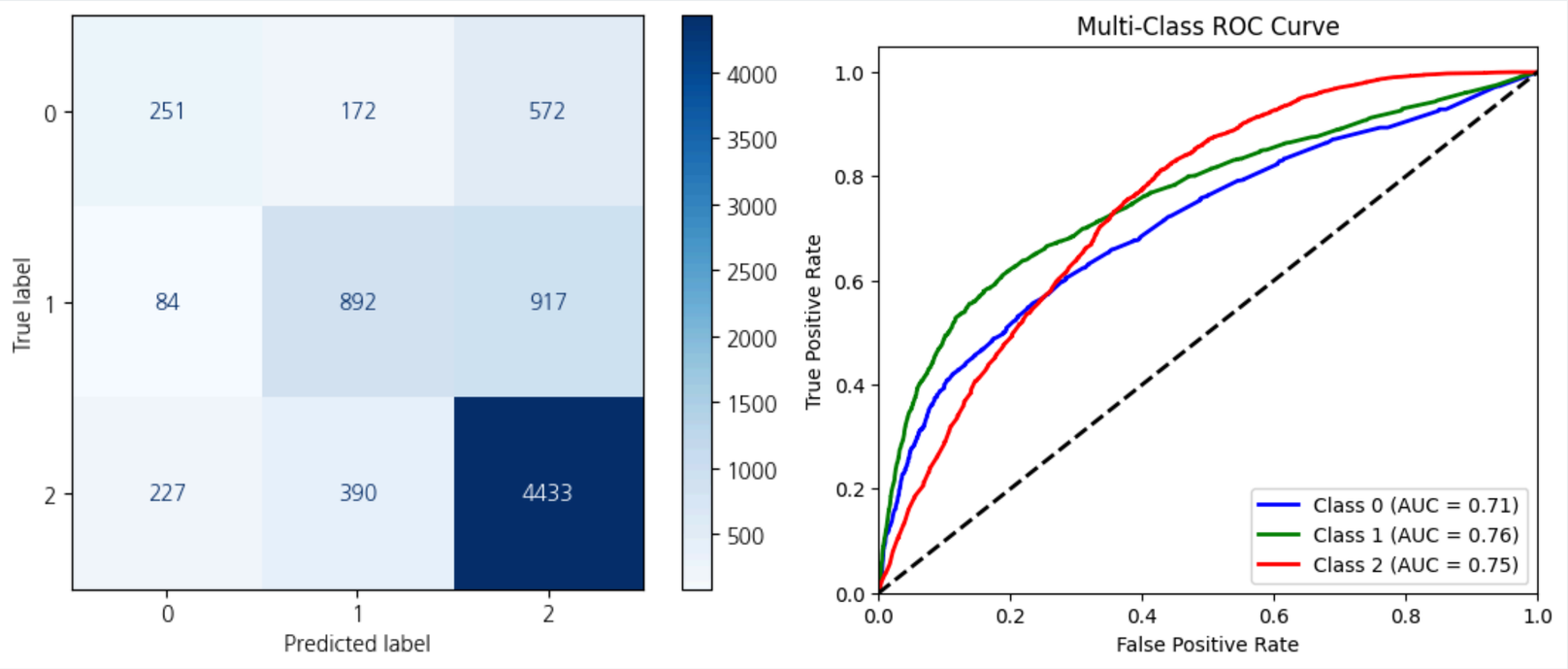


훈련 데이터 정확도 : 0.9805
검증 데이터 정확도 : 0.7016
F1 Score : 0.6781
AUC Score : 0.7322

4. 모델 학습 및 분석 결과

RandomForest

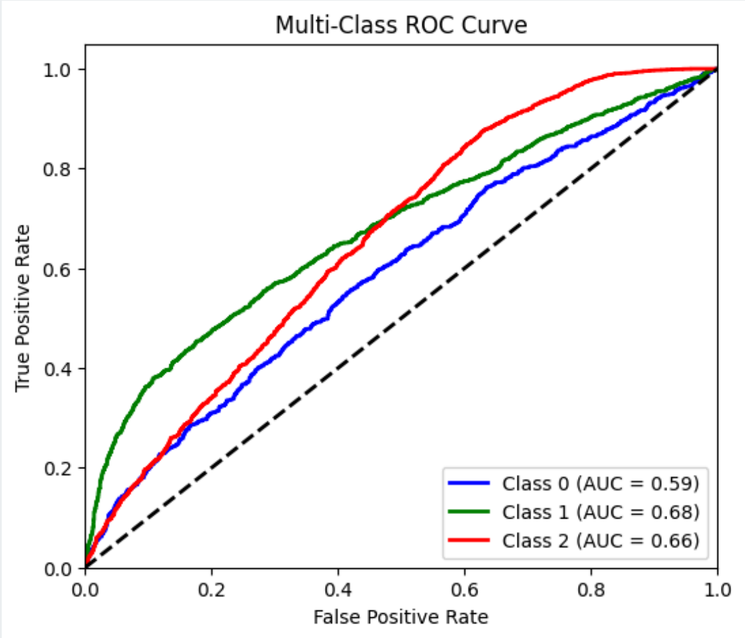
변수전체(라벨인코딩)



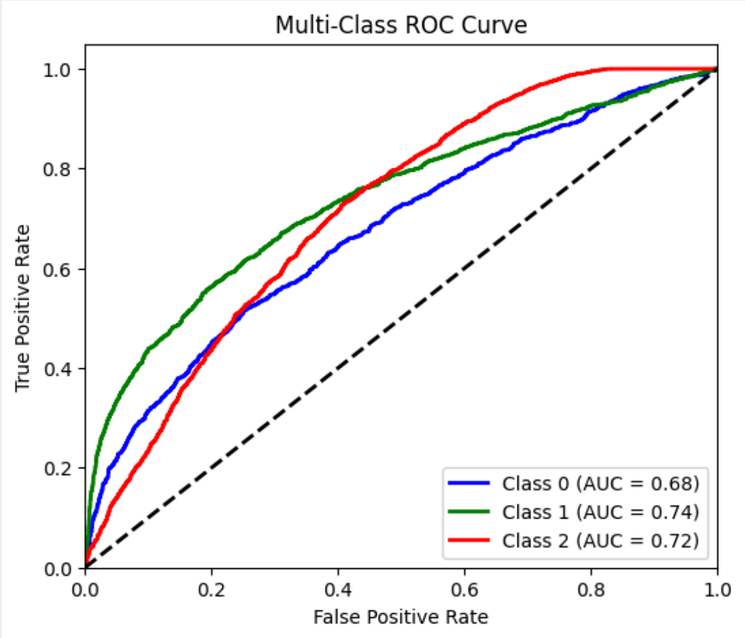
훈련 데이터 정확도 : 0.9806
검증 데이터 정확도 : 0.7024
F1 Score : 0.6816
AUC Score : 0.7396

4. 모델 학습 및 분석 결과

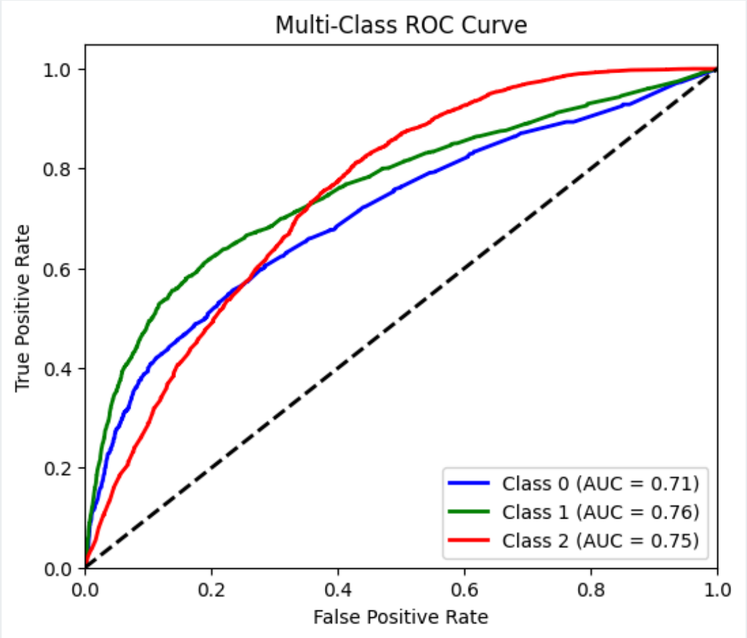
세 가지 모델 비교



DNN



XGBoost



RandomForest

5. 기대효과

신용카드 연체 예측의 기대효과

금융 기관의 리스크 관리

- 연체 위험 사전 파악
  : 대출 한도 조정
  : 신중한 신용 정책 수립

- 부실 채권 관리
  : 손실로 이어질 수 있는
  부실 채권 관리

맞춤형 금융 서비스

- 고객 맞춤형 서비스
  : 재무 상태에 맞는
  금융 상품 추천

- 고객 만족도 향상
  : 고객의 신용 점수를 유지

경제적 안정성

- 소비자 보호
  : 개인의 재무 안정성 유지

- 사회적 비용 감소
  : 대규모 연체 관리

딥러닝 AI개론

THANK YOU

빅데이터융합전공 202213212 박소현
빅데이터융합전공 202213419 이화연