In [12]:
```
import numpy as np
import pandas as pd
#pd.set_option("display.max_columns", 50)

import math

import matplotlib.pyplot as plt
import seaborn as sns

# from keras.models import Model, load_model
# from keras.layers import Input, Dense
# from keras.callbacks import ModelCheckpoint, TensorBoard, EarlyStopping
from keras import regularizers
import tensorflow_addons as tfa

from sklearn.preprocessing import LabelEncoder, StandardScaler
from sklearn.model_selection import train_test_split
from sklearn.metrics import *

import os

#GPU 자원이 부족
from tensorflow.compat.v1 import ConfigProto
from tensorflow.compat.v1 import InteractiveSession
config = ConfigProto()
config.gpu_options.allow_growth = True
session = InteractiveSession(config=config)
```

```
C:\Users\sjkan\anaconda3\lib\site-packages\tensorflow\python\client\session.py:1768: UserWarning: An interactiv
e session is already active. This can cause out-of-memory errors in some cases. You must explicitly call `Inter
activeSession.close()` to release resources held by the other session(s).
  warnings.warn('An interactive session is already active. This can '
```

RandomForest

In [48]:
```
df = pd.read_csv('C:/Users/sjkan/Downloads/tmp777.csv', encoding='utf-8')
```

In [49]:
```
df.drop(columns=['loanapply_insert_time','insert_time'],axis=1, inplace =True)
```

In [50]:
```
# 데이터 타입 변경

categorical_feats = ['bank_id','product_id','income_type', 'employment_type', 'houseown_type', 'purpose']

cat_df = df[categorical_feats]

for c in categorical_feats:
    df[c] = df[c].astype('category')

# 범주형 ordinal encoding

from category_encoders import OrdinalEncoder

enc1 = OrdinalEncoder(cols = cat_df.columns)

cat_df = enc1.fit_transform(cat_df)

df_tmp1 = df.drop(categorical_feats, axis = 1)
df_tmp1.reset_index(drop=True, inplace=True)
cat_df.reset_index(drop=True, inplace=True)
df = pd.concat([df_tmp1,cat_df], axis = 1)

del df_tmp1, cat_df
```

In [51]:
```
# 임시 1안
X = df.drop(['is_applied','application_id','user_id','bank_ratio'], axis=1) # 'bank_id',  'product_id'
Y = df['is_applied']
X_train, X_test, Y_train, Y_test = train_test_split(X, Y, test_size = 0.2, random_state = 42, stratify=Y)
```

In [52]:
```
# Building the Random Forest Classifier (RANDOM FOREST)
from sklearn.ensemble import RandomForestClassifier
# random forest model creation
rfc = RandomForestClassifier(random_state = 42, verbose = 2, n_jobs=-1, class_weight="balanced")
rfc.fit(X_train,Y_train)
```

```
[Parallel(n_jobs=-1)]: Using backend ThreadingBackend with 8 concurrent workers.
```

```
building tree 1 of 100building tree 2 of 100building tree 3 of 100

building tree 4 of 100
building tree 5 of 100
building tree 6 of 100

building tree 7 of 100
building tree 8 of 100
building tree 9 of 100
building tree 10 of 100
building tree 11 of 100
building tree 12 of 100
building tree 13 of 100
building tree 14 of 100
building tree 15 of 100
building tree 16 of 100
building tree 17 of 100
building tree 18 of 100
building tree 19 of 100
building tree 20 of 100
building tree 21 of 100
building tree 22 of 100
building tree 23 of 100
building tree 24 of 100
building tree 25 of 100
building tree 26 of 100
building tree 27 of 100
building tree 28 of 100
building tree 29 of 100
building tree 30 of 100
building tree 31 of 100
building tree 32 of 100
building tree 33 of 100building tree 34 of 100

[Parallel(n_jobs=-1)]: Done  25 tasks      | elapsed:  3.1min
```

```
building tree 35 of 100
building tree 36 of 100
building tree 37 of 100
building tree 38 of 100
building tree 39 of 100
building tree 40 of 100
building tree 41 of 100
building tree 42 of 100
building tree 43 of 100
building tree 44 of 100
building tree 45 of 100
building tree 46 of 100
building tree 47 of 100
building tree 48 of 100
building tree 49 of 100
building tree 50 of 100
building tree 51 of 100
building tree 52 of 100building tree 53 of 100

building tree 54 of 100
building tree 55 of 100
building tree 56 of 100
building tree 57 of 100
building tree 58 of 100
building tree 59 of 100
building tree 60 of 100
building tree 61 of 100
building tree 62 of 100
building tree 63 of 100
building tree 64 of 100
building tree 65 of 100
building tree 66 of 100
building tree 67 of 100
building tree 68 of 100
building tree 69 of 100
building tree 70 of 100
building tree 71 of 100
building tree 72 of 100
building tree 73 of 100
building tree 74 of 100
building tree 75 of 100
building tree 76 of 100
building tree 77 of 100
building tree 78 of 100
building tree 79 of 100
building tree 80 of 100
building tree 81 of 100
building tree 82 of 100
building tree 83 of 100
building tree 84 of 100
building tree 85 of 100
building tree 86 of 100
building tree 87 of 100

building tree 88 of 100
building tree 89 of 100
building tree 90 of 100
building tree 91 of 100
building tree 92 of 100
building tree 93 of 100
building tree 94 of 100
building tree 95 of 100
building tree 96 of 100
building tree 97 of 100
building tree 98 of 100
building tree 99 of 100
building tree 100 of 100
[Parallel(n_jobs=-1)]: Done 100 out of 100 | elapsed:  9.8min finished
```

Out[52]:

```
RandomForestClassifier(class_weight='balanced', n_jobs=-1, random_state=42,
                       verbose=2)
```

In [53]:

```
prob1 = rfc.predict_proba(X_test)
prob1
```

```
[Parallel(n_jobs=8)]: Using backend ThreadingBackend with 8 concurrent workers.
[Parallel(n_jobs=8)]: Done  25 tasks      | elapsed:    5.4s
[Parallel(n_jobs=8)]: Done 100 out of 100 | elapsed:   17.3s finished
```

Out[53]:

```
array([[1.        , 0.        ],
       [0.97      , 0.03      ],
       [0.99      , 0.01      ],
       ...,
       [0.99      , 0.01      ],
       [1.        , 0.        ],
       [0.94168237, 0.05831763]])
```

In [54]:

```
# user , product
```

```
result_col = []
for i in range(len(Y_test.values)):
        result_col.append(prob1[:,1][i])
```

In [55]:
```
# criteria를 바꾸면서 5.8%근처로
criteria = 0.259
result_bool = [0 if i <= criteria else 1 for i in result_col]

sum(result_bool)/len(result_bool) * 100
```

Out[55]:
```
5.884506650255287
```

In [56]:
```
y_true = Y_test
y_pred = result_bool
```

In [57]:
```
f1_score(y_true, y_pred, average='macro')
```

Out[57]:
```
0.7039447234259476
```

In [ ]:

In [ ]:

lgbm

In [13]:
```
df = pd.read_csv('C:/Users/sjkan/Downloads/tmp777.csv', encoding='utf-8')
df.head()
```

Out[13]:

	application_id	loanapply_insert_time	bank_id	product_id	loan_limit	loan_rate	is_applied	user_id	birth_year	gender	...	UsePrepayC
0	2157865	2022-05-09 08:44:59	54	235	20000000.0	16.5	1.0	346970.0	1970.0	1.0	...	
1	576643	2022-05-09 10:54:53	54	235	11000000.0	16.5	0.0	545882.0	1977.0	1.0	...	
2	576643	2022-05-09 10:54:53	11	118	3000000.0	20.0	0.0	545882.0	1977.0	1.0	...	
3	2136706	2022-05-09 10:41:06	42	216	10000000.0	13.5	0.0	558819.0	1983.0	1.0	...	
4	2136706	2022-05-09 10:41:07	25	169	22000000.0	15.9	0.0	558819.0	1983.0	1.0	...	

5 rows × 37 columns

In [14]:
```
df.drop(columns=['loanapply_insert_time','insert_time'],axis=1, inplace =True)
```

In [15]:
```
# 데이터 타입 변경

#df.info()

categorical_feats = ['bank_id','product_id','income_type', 'employment_type', 'houseown_type', 'purpose']

cat_df = df[categorical_feats]

for c in categorical_feats:
    df[c] = df[c].astype('category')
```

스케일링

In [16]:
```
from sklearn.preprocessing import RobustScaler, StandardScaler, MinMaxScaler

scaler_df = df.drop(['is_applied','bank_id','product_id','income_type', 'employment_type', 'houseown_type', 'pu

scale_df_col = scaler_df.columns

scaler = RobustScaler()
df_robust = scaler.fit_transform(scaler_df)

df_robust = pd.DataFrame(df_robust, columns = scale_df_col)


target = df[['is_applied','bank_id','product_id','income_type', 'employment_type', 'houseown_type', 'purpose']]
target.reset_index(drop=True, inplace=True)
afterscale_df = pd.concat([df_robust,target], axis = 1)
afterscale_df['is_applied'] = afterscale_df['is_applied'].astype('int')
```

In [17]:
```
X = afterscale_df.drop(['is_applied','application_id','user_id','bank_ratio'], axis=1) # 'bank_id',  'product_i
Y = afterscale_df['is_applied']
X_train, X_test, Y_train, Y_test = train_test_split(X, Y, test_size = 0.2, random_state = 42, stratify=Y)
```

In [22]:
```
from lightgbm import LGBMClassifier
```

```
rfc = LGBMClassifier(n_estimators = 200,
                     learning_rate=0.05,
                      # n_estimators 랑 같은 것 같음
                     max_depth = 16,
                    num_leaves = 80,
                     n_jobs=-1,
                     scale_pos_weight=5,
                     boosting_type='goss',
                     boost_from_average=False,
                     application  = 'binary',
                     force_col_wise=True,
                     verbose=2)

rfc.fit(X_train,Y_train)
```

```
[LightGBM] [Info] Number of positive: 301432, number of negative: 5073699
[LightGBM] [Debug] Dataset::GetMultiBinFromSparseFeatures: sparse rate 0.891217
[LightGBM] [Info] Total Bins 3768
[LightGBM] [Info] Number of data points in the train set: 5375131, number of used features: 31
[LightGBM] [Info] Using GOSS
[LightGBM] [Debug] Trained a tree with leaves = 80 and depth = 9
[LightGBM] [Debug] Trained a tree with leaves = 80 and depth = 9
[LightGBM] [Debug] Trained a tree with leaves = 80 and depth = 10
[LightGBM] [Debug] Trained a tree with leaves = 80 and depth = 9
[LightGBM] [Debug] Trained a tree with leaves = 80 and depth = 9
[LightGBM] [Debug] Trained a tree with leaves = 80 and depth = 9
[LightGBM] [Debug] Trained a tree with leaves = 80 and depth = 9
[LightGBM] [Debug] Trained a tree with leaves = 80 and depth = 9
[LightGBM] [Debug] Trained a tree with leaves = 80 and depth = 9
[LightGBM] [Debug] Trained a tree with leaves = 80 and depth = 10
[LightGBM] [Debug] Trained a tree with leaves = 80 and depth = 8
[LightGBM] [Debug] Trained a tree with leaves = 80 and depth = 9
[LightGBM] [Debug] Trained a tree with leaves = 80 and depth = 9
[LightGBM] [Debug] Trained a tree with leaves = 80 and depth = 10
[LightGBM] [Debug] Trained a tree with leaves = 80 and depth = 11
[LightGBM] [Debug] Trained a tree with leaves = 80 and depth = 9
[LightGBM] [Debug] Trained a tree with leaves = 80 and depth = 8
[LightGBM] [Debug] Trained a tree with leaves = 80 and depth = 10
[LightGBM] [Debug] Trained a tree with leaves = 80 and depth = 10
[LightGBM] [Debug] Trained a tree with leaves = 80 and depth = 10
[LightGBM] [Debug] Trained a tree with leaves = 80 and depth = 10
[LightGBM] [Debug] Trained a tree with leaves = 80 and depth = 10
[LightGBM] [Debug] Trained a tree with leaves = 80 and depth = 9
[LightGBM] [Debug] Trained a tree with leaves = 80 and depth = 9
[LightGBM] [Debug] Trained a tree with leaves = 80 and depth = 11
[LightGBM] [Debug] Trained a tree with leaves = 80 and depth = 12
[LightGBM] [Debug] Trained a tree with leaves = 80 and depth = 12
[LightGBM] [Debug] Trained a tree with leaves = 80 and depth = 10
[LightGBM] [Debug] Trained a tree with leaves = 80 and depth = 11
[LightGBM] [Debug] Trained a tree with leaves = 80 and depth = 11
[LightGBM] [Debug] Trained a tree with leaves = 80 and depth = 11
[LightGBM] [Debug] Trained a tree with leaves = 80 and depth = 11
[LightGBM] [Debug] Trained a tree with leaves = 80 and depth = 12
[LightGBM] [Debug] Trained a tree with leaves = 80 and depth = 10
[LightGBM] [Debug] Trained a tree with leaves = 80 and depth = 10
[LightGBM] [Debug] Trained a tree with leaves = 80 and depth = 10
[LightGBM] [Debug] Trained a tree with leaves = 80 and depth = 12
[LightGBM] [Debug] Trained a tree with leaves = 80 and depth = 10
[LightGBM] [Debug] Trained a tree with leaves = 80 and depth = 10
[LightGBM] [Debug] Trained a tree with leaves = 80 and depth = 11
[LightGBM] [Debug] Trained a tree with leaves = 80 and depth = 10
[LightGBM] [Debug] Trained a tree with leaves = 80 and depth = 10
[LightGBM] [Debug] Trained a tree with leaves = 80 and depth = 10
[LightGBM] [Debug] Trained a tree with leaves = 80 and depth = 11
[LightGBM] [Debug] Trained a tree with leaves = 80 and depth = 11
[LightGBM] [Debug] Trained a tree with leaves = 80 and depth = 14
[LightGBM] [Debug] Trained a tree with leaves = 80 and depth = 11
[LightGBM] [Debug] Trained a tree with leaves = 80 and depth = 12
[LightGBM] [Debug] Trained a tree with leaves = 80 and depth = 15
[LightGBM] [Debug] Trained a tree with leaves = 80 and depth = 12
[LightGBM] [Debug] Trained a tree with leaves = 80 and depth = 12
[LightGBM] [Debug] Trained a tree with leaves = 80 and depth = 14
[LightGBM] [Debug] Trained a tree with leaves = 80 and depth = 14
[LightGBM] [Debug] Trained a tree with leaves = 80 and depth = 15
[LightGBM] [Debug] Trained a tree with leaves = 80 and depth = 14
[LightGBM] [Debug] Trained a tree with leaves = 80 and depth = 11
[LightGBM] [Debug] Trained a tree with leaves = 80 and depth = 11
[LightGBM] [Debug] Trained a tree with leaves = 80 and depth = 13
[LightGBM] [Debug] Trained a tree with leaves = 80 and depth = 12
[LightGBM] [Debug] Trained a tree with leaves = 80 and depth = 13
[LightGBM] [Debug] Trained a tree with leaves = 80 and depth = 12
[LightGBM] [Debug] Trained a tree with leaves = 80 and depth = 11
[LightGBM] [Debug] Trained a tree with leaves = 80 and depth = 15
[LightGBM] [Debug] Trained a tree with leaves = 80 and depth = 11
[LightGBM] [Debug] Trained a tree with leaves = 80 and depth = 12
[LightGBM] [Debug] Trained a tree with leaves = 80 and depth = 14
[LightGBM] [Debug] Trained a tree with leaves = 80 and depth = 14
[LightGBM] [Debug] Trained a tree with leaves = 80 and depth = 13
```

```
[LightGBM] [Debug] Trained a tree with leaves = 80 and depth = 12
[LightGBM] [Debug] Trained a tree with leaves = 80 and depth = 16
[LightGBM] [Debug] Trained a tree with leaves = 80 and depth = 11
[LightGBM] [Debug] Trained a tree with leaves = 80 and depth = 16
[LightGBM] [Debug] Trained a tree with leaves = 80 and depth = 12
[LightGBM] [Debug] Trained a tree with leaves = 80 and depth = 14
[LightGBM] [Debug] Trained a tree with leaves = 80 and depth = 12
[LightGBM] [Debug] Trained a tree with leaves = 80 and depth = 16
[LightGBM] [Debug] Trained a tree with leaves = 80 and depth = 14
[LightGBM] [Debug] Trained a tree with leaves = 80 and depth = 16
[LightGBM] [Debug] Trained a tree with leaves = 80 and depth = 11
[LightGBM] [Debug] Trained a tree with leaves = 80 and depth = 15
[LightGBM] [Debug] Trained a tree with leaves = 80 and depth = 13
[LightGBM] [Debug] Trained a tree with leaves = 80 and depth = 14
[LightGBM] [Debug] Trained a tree with leaves = 80 and depth = 12
[LightGBM] [Debug] Trained a tree with leaves = 80 and depth = 13
[LightGBM] [Debug] Trained a tree with leaves = 80 and depth = 15
[LightGBM] [Debug] Trained a tree with leaves = 80 and depth = 14
[LightGBM] [Debug] Trained a tree with leaves = 80 and depth = 10
[LightGBM] [Debug] Trained a tree with leaves = 80 and depth = 16
[LightGBM] [Debug] Trained a tree with leaves = 80 and depth = 16
[LightGBM] [Debug] Trained a tree with leaves = 80 and depth = 13
[LightGBM] [Debug] Trained a tree with leaves = 80 and depth = 14
[LightGBM] [Debug] Trained a tree with leaves = 80 and depth = 12
[LightGBM] [Debug] Trained a tree with leaves = 80 and depth = 14
[LightGBM] [Debug] Trained a tree with leaves = 80 and depth = 16
[LightGBM] [Debug] Trained a tree with leaves = 80 and depth = 16
[LightGBM] [Debug] Trained a tree with leaves = 80 and depth = 13
[LightGBM] [Debug] Trained a tree with leaves = 80 and depth = 15
[LightGBM] [Debug] Trained a tree with leaves = 80 and depth = 16
[LightGBM] [Debug] Trained a tree with leaves = 80 and depth = 12
[LightGBM] [Debug] Trained a tree with leaves = 80 and depth = 13
[LightGBM] [Debug] Trained a tree with leaves = 80 and depth = 16
[LightGBM] [Debug] Trained a tree with leaves = 80 and depth = 12
[LightGBM] [Debug] Trained a tree with leaves = 80 and depth = 16
[LightGBM] [Debug] Trained a tree with leaves = 80 and depth = 12
[LightGBM] [Debug] Trained a tree with leaves = 80 and depth = 15
[LightGBM] [Debug] Trained a tree with leaves = 80 and depth = 14
[LightGBM] [Debug] Trained a tree with leaves = 80 and depth = 12
[LightGBM] [Debug] Trained a tree with leaves = 80 and depth = 16
[LightGBM] [Debug] Trained a tree with leaves = 80 and depth = 11
[LightGBM] [Debug] Trained a tree with leaves = 80 and depth = 12
[LightGBM] [Debug] Trained a tree with leaves = 80 and depth = 16
[LightGBM] [Debug] Trained a tree with leaves = 80 and depth = 16
[LightGBM] [Debug] Trained a tree with leaves = 80 and depth = 14
[LightGBM] [Debug] Trained a tree with leaves = 80 and depth = 15
[LightGBM] [Debug] Trained a tree with leaves = 80 and depth = 14
[LightGBM] [Debug] Trained a tree with leaves = 80 and depth = 15
[LightGBM] [Debug] Trained a tree with leaves = 80 and depth = 13
[LightGBM] [Debug] Trained a tree with leaves = 80 and depth = 15
[LightGBM] [Debug] Trained a tree with leaves = 80 and depth = 12
[LightGBM] [Debug] Trained a tree with leaves = 80 and depth = 12
[LightGBM] [Debug] Trained a tree with leaves = 80 and depth = 16
[LightGBM] [Debug] Trained a tree with leaves = 80 and depth = 16
[LightGBM] [Debug] Trained a tree with leaves = 80 and depth = 13
[LightGBM] [Debug] Trained a tree with leaves = 80 and depth = 14
[LightGBM] [Debug] Trained a tree with leaves = 80 and depth = 15
[LightGBM] [Debug] Trained a tree with leaves = 80 and depth = 16
[LightGBM] [Debug] Trained a tree with leaves = 80 and depth = 13
[LightGBM] [Debug] Trained a tree with leaves = 80 and depth = 13
[LightGBM] [Debug] Trained a tree with leaves = 80 and depth = 16
[LightGBM] [Debug] Trained a tree with leaves = 80 and depth = 16
[LightGBM] [Debug] Trained a tree with leaves = 80 and depth = 14
[LightGBM] [Debug] Trained a tree with leaves = 80 and depth = 16
[LightGBM] [Debug] Trained a tree with leaves = 80 and depth = 12
[LightGBM] [Debug] Trained a tree with leaves = 80 and depth = 13
[LightGBM] [Debug] Trained a tree with leaves = 80 and depth = 16
[LightGBM] [Debug] Trained a tree with leaves = 80 and depth = 12
[LightGBM] [Debug] Trained a tree with leaves = 80 and depth = 15
[LightGBM] [Debug] Trained a tree with leaves = 80 and depth = 13
[LightGBM] [Debug] Trained a tree with leaves = 80 and depth = 12
[LightGBM] [Debug] Trained a tree with leaves = 80 and depth = 12
[LightGBM] [Debug] Trained a tree with leaves = 80 and depth = 15
[LightGBM] [Debug] Trained a tree with leaves = 80 and depth = 16
[LightGBM] [Debug] Trained a tree with leaves = 80 and depth = 14
[LightGBM] [Debug] Trained a tree with leaves = 80 and depth = 13
[LightGBM] [Debug] Trained a tree with leaves = 80 and depth = 15
[LightGBM] [Debug] Trained a tree with leaves = 80 and depth = 12
[LightGBM] [Debug] Trained a tree with leaves = 80 and depth = 14
[LightGBM] [Debug] Trained a tree with leaves = 80 and depth = 13
[LightGBM] [Debug] Trained a tree with leaves = 80 and depth = 16
[LightGBM] [Debug] Trained a tree with leaves = 80 and depth = 13
[LightGBM] [Debug] Trained a tree with leaves = 80 and depth = 15
[LightGBM] [Debug] Trained a tree with leaves = 80 and depth = 15
[LightGBM] [Debug] Trained a tree with leaves = 80 and depth = 14
[LightGBM] [Debug] Trained a tree with leaves = 80 and depth = 16
[LightGBM] [Debug] Trained a tree with leaves = 80 and depth = 16
[LightGBM] [Debug] Trained a tree with leaves = 80 and depth = 14
[LightGBM] [Debug] Trained a tree with leaves = 80 and depth = 14
```

```
[LightGBM] [Debug] Trained a tree with leaves = 80 and depth = 13
[LightGBM] [Debug] Trained a tree with leaves = 80 and depth = 13
[LightGBM] [Debug] Trained a tree with leaves = 80 and depth = 15
[LightGBM] [Debug] Trained a tree with leaves = 80 and depth = 16
[LightGBM] [Debug] Trained a tree with leaves = 80 and depth = 14
[LightGBM] [Debug] Trained a tree with leaves = 80 and depth = 14
[LightGBM] [Debug] Trained a tree with leaves = 80 and depth = 15
[LightGBM] [Debug] Trained a tree with leaves = 80 and depth = 14
[LightGBM] [Debug] Trained a tree with leaves = 80 and depth = 12
[LightGBM] [Debug] Trained a tree with leaves = 80 and depth = 15
[LightGBM] [Debug] Trained a tree with leaves = 80 and depth = 15
[LightGBM] [Debug] Trained a tree with leaves = 80 and depth = 13
[LightGBM] [Debug] Trained a tree with leaves = 80 and depth = 13
[LightGBM] [Debug] Trained a tree with leaves = 80 and depth = 15
[LightGBM] [Debug] Trained a tree with leaves = 80 and depth = 16
[LightGBM] [Debug] Trained a tree with leaves = 80 and depth = 16
[LightGBM] [Debug] Trained a tree with leaves = 80 and depth = 15
[LightGBM] [Debug] Trained a tree with leaves = 80 and depth = 16
[LightGBM] [Debug] Trained a tree with leaves = 80 and depth = 15
[LightGBM] [Debug] Trained a tree with leaves = 80 and depth = 14
[LightGBM] [Debug] Trained a tree with leaves = 80 and depth = 13
[LightGBM] [Debug] Trained a tree with leaves = 80 and depth = 11
[LightGBM] [Debug] Trained a tree with leaves = 80 and depth = 16
[LightGBM] [Debug] Trained a tree with leaves = 80 and depth = 14
[LightGBM] [Debug] Trained a tree with leaves = 80 and depth = 16
[LightGBM] [Debug] Trained a tree with leaves = 80 and depth = 14
[LightGBM] [Debug] Trained a tree with leaves = 80 and depth = 14
[LightGBM] [Debug] Trained a tree with leaves = 80 and depth = 12
[LightGBM] [Debug] Trained a tree with leaves = 80 and depth = 13
[LightGBM] [Debug] Trained a tree with leaves = 80 and depth = 16
[LightGBM] [Debug] Trained a tree with leaves = 80 and depth = 16
[LightGBM] [Debug] Trained a tree with leaves = 80 and depth = 16
[LightGBM] [Debug] Trained a tree with leaves = 80 and depth = 16
[LightGBM] [Debug] Trained a tree with leaves = 80 and depth = 12
[LightGBM] [Debug] Trained a tree with leaves = 80 and depth = 16
[LightGBM] [Debug] Trained a tree with leaves = 80 and depth = 16
[LightGBM] [Debug] Trained a tree with leaves = 80 and depth = 16
[LightGBM] [Debug] Trained a tree with leaves = 80 and depth = 16
[LightGBM] [Debug] Trained a tree with leaves = 80 and depth = 13
[LightGBM] [Debug] Trained a tree with leaves = 80 and depth = 16
[LightGBM] [Debug] Trained a tree with leaves = 80 and depth = 16
[LightGBM] [Debug] Trained a tree with leaves = 80 and depth = 13
[LightGBM] [Debug] Trained a tree with leaves = 80 and depth = 16
```

Out[22]:

```
LGBMClassifier(application='binary', boost_from_average=False,
               boosting_type='goss', force_col_wise=True, learning_rate=0.05,
               max_depth=16, n_estimators=200, num_leaves=80,
               scale_pos_weight=5, verbose=2)
```

In [23]:
```
prob2 = rfc.predict_proba(X_test)
prob2
```

Out[23]:
```
array([[0.98021753, 0.01978247],
       [0.8892925 , 0.1107075 ],
       [0.94623517, 0.05376483],
       ...,
       [0.97068182, 0.02931818],
       [0.96840177, 0.03159823],
       [0.68563322, 0.31436678]])
```

In [24]:
```
result_col = []
for i in range(len(Y_test.values)):
    result_col.append(prob2[:,1][i])
```

In [25]:
```
# criteria를 바꾸면서 6%근처로
criteria = 0.6555
result_bool = [0 if i <= criteria else 1 for i in result_col]

sum(result_bool)/len(result_bool) * 100
```

Out[25]:
```
5.796397186152824
```

In [26]:
```
y_true = Y_test
y_pred = result_bool
```

In [27]:
```
f1_score(y_true, y_pred, average='macro')
```

Out[27]:
```
0.7053674600060931
```

In [ ]:

In [ ]:

제출 파일 만들어서 비교

In [47]:
```
df2 = pd.read_csv('C:/Users/sjkan/Downloads/rawdata_tmp3.csv', encoding='utf-8')
```

In [48]:
```
df2.drop(columns=['personal_rehabilitation_yn',
       'personal_rehabilitation_complete_yn'],axis=1, inplace =True)
```

In [49]:
```
df2 = df2[df2['is_applied'].isnull()==True]
```

In [50]:
```
df2['credit_score'].fillna(759, inplace=True)
```

In [51]:
```
df2.drop(columns=['loanapply_insert_time','insert_time'],axis=1, inplace =True)
categorical_feats = ['bank_id','product_id','income_type', 'employment_type', 'houseown_type', 'purpose']
cat_df = df2[categorical_feats]

for c in categorical_feats:
    df2[c] = df2[c].astype('category')

from sklearn.preprocessing import RobustScaler, StandardScaler, MinMaxScaler

scaler_df = df2.drop(['is_applied','bank_id','product_id','income_type', 'employment_type', 'houseown_type', 'p
scale_df_col = scaler_df.columns

scaler = RobustScaler()
df_robust = scaler.fit_transform(scaler_df)
df_robust = pd.DataFrame(df_robust, columns = scale_df_col)

target = df2[['is_applied','bank_id','product_id','income_type', 'employment_type', 'houseown_type', 'purpose']
target.reset_index(drop=True, inplace=True)
afterscale_df = pd.concat([df_robust,target], axis = 1)
#afterscale_df['is_applied'] = afterscale_df['is_applied'].astype('int')
```

In [52]:
```
df6 = afterscale_df[afterscale_df['is_applied'].isnull()==True]
x_test_june = df6.drop(columns=['is_applied','application_id','user_id'], axis=1)
y_test_june = df6['is_applied']
```

In [53]:
```
answer = rfc.predict_proba(x_test_june)
```

In [54]:
```
result_col = []
for i in range(len(y_test_june.values)):
    result_col.append(answer[:,1][i])
```

In [55]:
```
# criteria를 바꾸면서 6%근처로
criteria = 0.6555
result_bool = [0 if i <= criteria else 1 for i in result_col]

sum(result_bool)/len(result_bool) * 100
```

Out[55]:
```
8.106664389482109
```

In [59]:
```
submission = df2[['application_id','product_id','is_applied']]
```

In [63]:
```
submission['is_applied']=result_bool
```

```
C:\Users\sjkan\AppData\Local\Temp\ipykernel_14604\2026824180.py:1: SettingWithCopyWarning:
A value is trying to be set on a copy of a slice from a DataFrame.
Try using .loc[row_indexer,col_indexer] = value instead

See the caveats in the documentation: https://pandas.pydata.org/pandas-docs/stable/user_guide/indexing.html#ret
urning-a-view-versus-a-copy
  submission['is_applied']=result_bool
```

In [64]:
```
submission
```

Out[64]:

	application_id	product_id	is_applied
0	1748340	191	0
1	1748340	169	0
2	1748340	7	0
3	1748340	268	1
4	1748340	118	0
...	...	...	...
13519863	1428218	200	0
13519864	1428218	7	0
13519865	1428218	257	0
13519866	1428218	110	0
13519867	1428218	194	0

3255482 rows × 3 columns

In [69]:
```
submit = pd.read_csv('C:/Users/sjkan/Downloads/데이터분석분야_퓨처스부문_평가데이터.csv', encoding='utf-8', usecols=[
```

In [70]:
```
submit
```

Out[70]:

	application_id	product_id
0	4	220
1	4	191
2	8	29
3	8	159
4	8	85
...	...	...
3255189	2167778	258
3255190	2167791	29
3255191	2167822	149
3255192	2167822	157
3255193	2167822	65

3255194 rows × 2 columns

In [72]:
```
# 기준열 이름이 같을 때
real_answer = pd.merge(submit, submission, on = ['application_id','product_id'], how = 'left')
```

In [75]:
```
real_answer
```

Out[75]:

	application_id	product_id	is_applied
0	4	220	0
1	4	191	1
2	8	29	1
3	8	159	0
4	8	85	1
...	...	...	...
3255189	2167778	258	1
3255190	2167791	29	1
3255191	2167822	149	1
3255192	2167822	157	1
3255193	2167822	65	0

3255194 rows × 3 columns

In [74]:
```
#real_answer.to_csv("./data들/와빅병아리-코드결과.csv", encoding='utf-8', index=False)
```

In [ ]:

In [ ]:

Xgboost

Xgboost

In [138…
```
df = pd.read_csv('C:/Users/sjkan/Downloads/tmp777.csv', encoding='utf-8')
df.head()
```

Out[138]:

	application_id	loanapply_insert_time	bank_id	product_id	loan_limit	loan_rate	is_applied	user_id	birth_year	gender	...	UsePrepay(
0	2157865	2022-05-09 08:44:59	54	235	20000000.0	16.5	1.0	346970.0	1970.0	1.0	...	
1	576643	2022-05-09 10:54:53	54	235	11000000.0	16.5	0.0	545882.0	1977.0	1.0	...	
2	576643	2022-05-09 10:54:53	11	118	3000000.0	20.0	0.0	545882.0	1977.0	1.0	...	
3	2136706	2022-05-09 10:41:06	42	216	10000000.0	13.5	0.0	558819.0	1983.0	1.0	...	
4	2136706	2022-05-09 10:41:07	25	169	22000000.0	15.9	0.0	558819.0	1983.0	1.0	...	

5 rows × 37 columns

In [139…
```
df.drop(columns=['loanapply_insert_time','insert_time'],axis=1, inplace =True)
```

In [140…
```
# 데이터 타입 변경

#df.info()

categorical_feats = ['bank_id','product_id','income_type', 'employment_type', 'houseown_type', 'purpose']

cat_df = df[categorical_feats]

for c in categorical_feats:
    df[c] = df[c].astype('category')

# 범주형 ordinal encoding

from category_encoders import OrdinalEncoder

enc1 = OrdinalEncoder(cols = cat_df.columns)

#cat_df

cat_df = enc1.fit_transform(cat_df)
#cat_df.head()

df_tmp1 = df.drop(categorical_feats, axis = 1)
df_tmp1.reset_index(drop=True, inplace=True)
cat_df.reset_index(drop=True, inplace=True)
df = pd.concat([df_tmp1,cat_df], axis = 1)

del df_tmp1, cat_df
```

스케일링

In [141…
```
from sklearn.preprocessing import RobustScaler, StandardScaler, MinMaxScaler


scaler_df = df.drop(['is_applied'], axis = 1)

scale_df_col = scaler_df.columns

scaler = RobustScaler()
df_robust = scaler.fit_transform(scaler_df)

df_robust = pd.DataFrame(df_robust, columns = scale_df_col)


target = df[['is_applied']]

target.reset_index(drop=True, inplace=True)

afterscale_df = pd.concat([df_robust,target], axis = 1)

afterscale_df['is_applied'] = afterscale_df['is_applied'].astype('int')

afterscale_df.head()
```

Out[141]:

	application_id	loan_limit	loan_rate	user_id	birth_year	gender	credit_score	yearly_income	company_enter_month	desired_amount	.
0	0.989750	0.130435	0.508197	-0.213592	-0.928571	0.0	-0.947368	-0.25	-1.395960	-0.375	.
1	-0.468646	-0.260870	0.508197	0.239626	-0.428571	0.0	-0.736842	1.75	-0.191919	-0.125	.
2	-0.468646	-0.608696	1.081967	0.239626	-0.428571	0.0	-0.736842	1.75	-0.191919	-0.125	.
3	0.970234	-0.304348	0.016393	0.269102	0.000000	0.0	0.105263	0.10	-1.616162	1.375	.
4	0.970234	0.217391	0.409836	0.269102	0.000000	0.0	0.105263	0.10	-1.616162	1.375	.

5 rows × 35 columns

In [ ]:

In [142…
```
#1안 - bank_ratio 일단 빼기
X = df.drop(['is_applied','application_id','user_id','bank_ratio'], axis=1) # 'bank_id',  'product_id'
Y = df['is_applied']
X_train, X_test, Y_train, Y_test = train_test_split(X, Y, test_size = 0.2, random_state = 42, stratify=Y)
```

In [188…
```
from xgboost import plot_importance, XGBClassifier
rfc = XGBClassifier(random_state = 42, n_jobs=-1, scale_pos_weight=5, n_estimators = 100)
evals = [(X_test, Y_test)]
rfc.fit(X_train, Y_train, early_stopping_rounds = 3, eval_metric = "logloss", eval_set = evals, verbose = 2)
```

```
C:\Users\sjkan\anaconda3\lib\site-packages\xgboost\sklearn.py:793: UserWarning: `eval_metric` in `fit` method i
s deprecated for better compatibility with scikit-learn, use `eval_metric` in constructor or`set_params` instea
d.
  warnings.warn(
C:\Users\sjkan\anaconda3\lib\site-packages\xgboost\sklearn.py:793: UserWarning: `early_stopping_rounds` in `fit
` method is deprecated for better compatibility with scikit-learn, use `early_stopping_rounds` in constructor o
r`set_params` instead.
  warnings.warn(
[0]     validation_0-logloss:0.52383
[2]     validation_0-logloss:0.36650
[4]     validation_0-logloss:0.29736
[6]     validation_0-logloss:0.26355
[8]     validation_0-logloss:0.24530
[10]    validation_0-logloss:0.23501
[12]    validation_0-logloss:0.22846
[14]    validation_0-logloss:0.22416
[16]    validation_0-logloss:0.22090
[18]    validation_0-logloss:0.21891
[20]    validation_0-logloss:0.21704
[22]    validation_0-logloss:0.21582
[24]    validation_0-logloss:0.21471
[26]    validation_0-logloss:0.21378
[28]    validation_0-logloss:0.21305
[30]    validation_0-logloss:0.21217
[32]    validation_0-logloss:0.21165
[34]    validation_0-logloss:0.21124
[36]    validation_0-logloss:0.21066
[38]    validation_0-logloss:0.21019
[40]    validation_0-logloss:0.20972
[42]    validation_0-logloss:0.20934
[44]    validation_0-logloss:0.20895
[46]    validation_0-logloss:0.20867
[48]    validation_0-logloss:0.20842
[50]    validation_0-logloss:0.20813
[52]    validation_0-logloss:0.20794
[54]    validation_0-logloss:0.20774
[56]    validation_0-logloss:0.20759
[58]    validation_0-logloss:0.20735
[60]    validation_0-logloss:0.20711
[62]    validation_0-logloss:0.20693
[64]    validation_0-logloss:0.20680
[66]    validation_0-logloss:0.20667
[68]    validation_0-logloss:0.20645
[70]    validation_0-logloss:0.20625
[72]    validation_0-logloss:0.20613
[74]    validation_0-logloss:0.20595
[76]    validation_0-logloss:0.20572
[78]    validation_0-logloss:0.20559
[80]    validation_0-logloss:0.20547
[82]    validation_0-logloss:0.20528
[84]    validation_0-logloss:0.20513
[86]    validation_0-logloss:0.20501
[88]    validation_0-logloss:0.20495
[90]    validation_0-logloss:0.20481
[92]    validation_0-logloss:0.20475
[94]    validation_0-logloss:0.20460
[96]    validation_0-logloss:0.20448
[98]    validation_0-logloss:0.20436
[99]    validation_0-logloss:0.20430
```

Out[188]:

```
XGBClassifier(base_score=0.5, booster='gbtree', callbacks=None,
              colsample_bylevel=1, colsample_bynode=1, colsample_bytree=1,
              early_stopping_rounds=None, enable_categorical=False,
              eval_metric=None, gamma=0, gpu_id=-1, grow_policy='depthwise',
              importance_type=None, interaction_constraints='',
              learning_rate=0.300000012, max_bin=256, max_cat_to_onehot=4,
              max_delta_step=0, max_depth=6, max_leaves=0, min_child_weight=1,
              missing=nan, monotone_constraints='()', n_estimators=100,
              n_jobs=-1, num_parallel_tree=1, predictor='auto', random_state=42,
              reg_alpha=0, reg_lambda=1, ...)
```

In [189…
```
prob3 = rfc.predict_proba(X_test)
prob3
```

Out[189]:
```
array([[0.97937477, 0.02062526],
       [0.84854066, 0.1514593 ],
       [0.95423937, 0.04576062],
       ...,
       [0.98258114, 0.01741884],
       [0.9386003 , 0.0613997 ],
       [0.6191329 , 0.38086712]], dtype=float32)
```

In [199…
```
result_col = []
for i in range(len(Y_test.values)):
        result_col.append(prob3[:,1][i])
```

In [205…
```
# criteria를 바꾸면서 6%근처로
criteria = 0.6635
result_bool = [0 if i <= criteria else 1 for i in result_col]

sum(result_bool)/len(result_bool) * 100
```

Out[205]:
```
5.794164682839416
```

In [206…
```
y_true = Y_test
y_pred = result_bool
```

In [207…
```
f1_score(y_true, y_pred, average='macro')
```

Out[207]:
```
0.7025932342764355
```

In [ ]:
```

```

In [213…
```
#2안 - bank_ratio 넣기
X = df.drop(['is_applied','application_id','user_id'], axis=1) # 'bank_id',  'product_id'
Y = df['is_applied']
X_train, X_test, Y_train, Y_test = train_test_split(X, Y, test_size = 0.2, random_state = 42, stratify=Y)
```

In [214…
```
from xgboost import plot_importance, XGBClassifier
rfc = XGBClassifier(random_state = 42, n_jobs=-1, scale_pos_weight=5, n_estimators = 100)
evals = [(X_test, Y_test)]
rfc.fit(X_train, Y_train, early_stopping_rounds = 3, eval_metric = "logloss", eval_set = evals, verbose = 2)
```

```
C:\Users\sjkan\anaconda3\lib\site-packages\xgboost\sklearn.py:793: UserWarning: `eval_metric` in `fit` method i
s deprecated for better compatibility with scikit-learn, use `eval_metric` in constructor or`set_params` instea
d.
  warnings.warn(
C:\Users\sjkan\anaconda3\lib\site-packages\xgboost\sklearn.py:793: UserWarning: `early_stopping_rounds` in `fit
` method is deprecated for better compatibility with scikit-learn, use `early_stopping_rounds` in constructor o
r`set_params` instead.
  warnings.warn(
```

```
[0]     validation_0-logloss:0.52370
[2]     validation_0-logloss:0.36563
[4]     validation_0-logloss:0.29604
[6]     validation_0-logloss:0.26185
[8]     validation_0-logloss:0.24356
[10]    validation_0-logloss:0.23294
[12]    validation_0-logloss:0.22670
[14]    validation_0-logloss:0.22256
[16]    validation_0-logloss:0.21964
[18]    validation_0-logloss:0.21776
[20]    validation_0-logloss:0.21614
[22]    validation_0-logloss:0.21509
[24]    validation_0-logloss:0.21413
[26]    validation_0-logloss:0.21314
[28]    validation_0-logloss:0.21239
[30]    validation_0-logloss:0.21168
[32]    validation_0-logloss:0.21097
[34]    validation_0-logloss:0.21051

[36]    validation_0-logloss:0.21004
[38]    validation_0-logloss:0.20962
[40]    validation_0-logloss:0.20930
[42]    validation_0-logloss:0.20896
[44]    validation_0-logloss:0.20874
[46]    validation_0-logloss:0.20841
[48]    validation_0-logloss:0.20809
[50]    validation_0-logloss:0.20783
[52]    validation_0-logloss:0.20753
[54]    validation_0-logloss:0.20724
[56]    validation_0-logloss:0.20702
[58]    validation_0-logloss:0.20680
[60]    validation_0-logloss:0.20665
[62]    validation_0-logloss:0.20643
[64]    validation_0-logloss:0.20624
[66]    validation_0-logloss:0.20609
[68]    validation_0-logloss:0.20599
[70]    validation_0-logloss:0.20588
[72]    validation_0-logloss:0.20576
[74]    validation_0-logloss:0.20555
[76]    validation_0-logloss:0.20533
[78]    validation_0-logloss:0.20517
[80]    validation_0-logloss:0.20506
[82]    validation_0-logloss:0.20490
[84]    validation_0-logloss:0.20484
[86]    validation_0-logloss:0.20473
[88]    validation_0-logloss:0.20463
[90]    validation_0-logloss:0.20448
[92]    validation_0-logloss:0.20438
[94]    validation_0-logloss:0.20418
[96]    validation_0-logloss:0.20408
[98]    validation_0-logloss:0.20396
[99]    validation_0-logloss:0.20387
```

Out[214]:

▾ XGBClassifier

```
XGBClassifier(base_score=0.5, booster='gbtree', callbacks=None,
              colsample_bylevel=1, colsample_bynode=1, colsample_bytree=1,
              early_stopping_rounds=None, enable_categorical=False,
              eval_metric=None, gamma=0, gpu_id=-1, grow_policy='depthwise',
              importance_type=None, interaction_constraints='',
              learning_rate=0.300000012, max_bin=256, max_cat_to_onehot=4,
              max_delta_step=0, max_depth=6, max_leaves=0, min_child_weight=1,
              missing=nan, monotone_constraints='()', n_estimators=100,
              n_jobs=-1, num_parallel_tree=1, predictor='auto', random_state=42,
              reg_alpha=0, reg_lambda=1, ...)
```

In [215…

```
prob4 = rfc.predict_proba(X_test)
prob4
```

Out[215]:

```
array([[0.97838175, 0.02161827],
       [0.86386824, 0.13613175],
       [0.961997  , 0.03800301],
       ...,
       [0.9878017 , 0.01219834],
       [0.9523356 , 0.04766443],
       [0.678995  , 0.32100502]], dtype=float32)
```

In [216…

```
# user , product
result_col = []
for i in range(len(Y_test.values)):
        result_col.append(prob4[:,1][i])
```

In [222…

```
criteria = 0.663
result_bool = [0 if i <= criteria else 1 for i in result_col]

sum(result_bool)/len(result_bool) * 100
```

Out[222]: 5.794908850610552

In [223…
```
y_true = Y_test
y_pred = result_bool
```

In [224…
```
f1_score(y_true, y_pred, average='macro')
```

Out[224]: 0.7034973274606185

In [ ]:

In [ ]:

lgbm+xgboost

prob1 : randomforest prob2 : lgbm prob3,4 : xgboose

In [225…
```
result_col = []
for i in range(len(Y_test.values)):
        result_col.append(prob2[:,1][i]*0.8 + prob3[:,1][i] *0.2)
```

In [226…
```
# criteria를 바꾸면서 6%근처로
criteria = 0.654
result_bool = [0 if i <= criteria else 1 for i in result_col]

sum(result_bool)/len(result_bool) * 100
```

Out[226]: 5.799299440460253

In [227…
```
y_true = Y_test
y_pred = result_bool
```

In [228…
```
f1_score(y_true, y_pred, average='macro')
```

Out[228]: 0.7062161923449911

In [ ]: