08 분류실습 - 캐글 산탄데르 고객 만족 예측

고객 만족 데이터 세트에 대해 고객 만족 여부를 XGBoost와 LightGBM을 활용해 예측

값이 1이면 불만을 가진 고객, 0이면 만족한 고객

```
#데이터 전처리
import numpy as np
import pandas as pd
import matplotlib.pyplot as plt
import matplotlib

cust_df = pd.read_csv("train_santander.csv",encoding='latin-1')
print('dataset shape:', cust_df.shape)
cust_df.head(3)
```

```
cust_df.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 76020 entries, 0 to 76019
Columns: 371 entries, ID to TARGET
dtypes: float64(111), int64(260)
memory usage: 215.2 MB
```

```
#만족과 불만족의 비율
print(cust_df['TARGET'].value_counts())
unsatisfied_cnt = cust_df[cust_df['TARGET'] == 1].TARGET.count()
total_cnt = cust_df.TARGET.count()
print('unsatisfied 비율은 {0:.2f}'.format((unsatisfied_cnt / total_cnt)))
```

```
0    73012
1     3008
Name: TARGET, dtype: int64
unsatisfied 비율은 0.04
```

```
cust_df.describe( )
```

```
# var3 피처 값 -999999를 가장 값이 많은 2로 대체
cust_df['var3'].replace(-999999,2, inplace=True)

# 단순 식별자 ID 드롭
cust_df.drop('ID',axis=1 , inplace=True)

# 피처 세트와 레이블 세트분리. 레이블 컬럼은 DataFrame의 맨 마지막에 위치해 컬럼 위치 -1로 분리
X_features = cust_df.iloc[:, :-1]
y_labels = cust_df.iloc[:, -1]
print('피처 데이터 shape:{0}'.format(X_features.shape))
```

```
피처 데이터 shape:(76020, 369)
```

```
#학습데이터와 테스트 데이터 세트 분리(target 값 분포도가 비슷하게 추출됐는지 확인)
from sklearn.model_selection import train_test_split

X_train, X_test, y_train, y_test = train_test_split(X_features, y_labels,
                                                    test_size=0.2, random_state=0)
train_cnt = y_train.count()
test_cnt = y_test.count()
print('학습 세트 Shape:{0}, 테스트 세트 Shape:{1}'.format(X_train.shape , X_test.shape))

print(' 학습 세트 레이블 값 분포 비율')
print(y_train.value_counts()/train_cnt)
print('₩n 테스트 세트 레이블 값 분포 비율')
print(y_test.value_counts()/test_cnt)
```

```
학습 세트 Shape:(60816, 369), 테스트 세트 Shape:(15204, 369)
 학습 세트 레이블 값 분포 비율
0    0.960964
1    0.039036
```

```
Name: TARGET, dtype: float64

 테스트 세트 레이블 값 분포 비율
0    0.9583
1    0.0417
Name: TARGET, dtype: float64
```

XGBoost 모델 학습과 하이퍼 파라미터 튜닝

```
#XGBoost 모델 학습
from xgboost import XGBClassifier
from sklearn.metrics import roc_auc_score

# n_estimators는 500으로, random state는 예제 수행 시마다 동일 예측 결과를 위해 설정
xgb_clf = XGBClassifier(n_estimators=500, random_state=156)

# 성능 평가 지표를 auc로, 조기 중단 파라미터는 100으로 설정하고 학습 수행
xgb_clf.fit(X_train, y_train, early_stopping_rounds=100,
            eval_metric="auc", eval_set=[(X_train, y_train), (X_test, y_test)])

xgb_roc_score = roc_auc_score(y_test, xgb_clf.predict_proba(X_test)[:,1],average='macro')
print('ROC AUC: {0:.4f}'.format(xgb_roc_score))
```

```
[114]   validation_0-auc:0.858382       validation_1-auc:0.841204
[115]   validation_0-auc:0.85846        validation_1-auc:0.841295
[116]   validation_0-auc:0.858562       validation_1-auc:0.841256
[117]   validation_0-auc:0.858616       validation_1-auc:0.84133
[118]   validation_0-auc:0.858718       validation_1-auc:0.841398
[119]   validation_0-auc:0.858829       validation_1-auc:0.841496
[120]   validation_0-auc:0.859012       validation_1-auc:0.841352
[121]   validation_0-auc:0.859142       validation_1-auc:0.841349
[122]   validation_0-auc:0.859203       validation_1-auc:0.841364
[123]   validation_0-auc:0.859284       validation_1-auc:0.84141
[124]   validation_0-auc:0.859563       validation_1-auc:0.841393
[125]   validation_0-auc:0.859639       validation_1-auc:0.84149
[126]   validation_0-auc:0.859759       validation_1-auc:0.841484
[127]   validation_0-auc:0.859957       validation_1-auc:0.841333
[128]   validation_0-auc:0.860087       validation_1-auc:0.841352
[129]   validation_0-auc:0.860211       validation_1-auc:0.841287
[130]   validation_0-auc:0.860261       validation_1-auc:0.841392
[131]   validation_0-auc:0.860444       validation_1-auc:0.841481
[132]   validation_0-auc:0.860453       validation_1-auc:0.841462
[133]   validation_0-auc:0.860607       validation_1-auc:0.841342
[134]   validation_0-auc:0.860749       validation_1-auc:0.84124
[135]   validation_0-auc:0.86094        validation_1-auc:0.841318
[136]   validation_0-auc:0.861021       validation_1-auc:0.841351
[137]   validation_0-auc:0.861131       validation_1-auc:0.841311
[138]   validation_0-auc:0.861229       validation_1-auc:0.841289
[139]   validation_0-auc:0.861279       validation_1-auc:0.841295
[140]   validation_0-auc:0.861331       validation_1-auc:0.841265
[141]   validation_0-auc:0.861418       validation_1-auc:0.841259
[142]   validation_0-auc:0.861553       validation_1-auc:0.841335
[143]   validation_0-auc:0.861682       validation_1-auc:0.841346
[144]   validation_0-auc:0.86169        validation_1-auc:0.841403
[145]   validation_0-auc:0.861852       validation_1-auc:0.841299
[146]   validation_0-auc:0.861898       validation_1-auc:0.841301
[147]   validation_0-auc:0.861998       validation_1-auc:0.841289
```

```
[148]   validation_0-auc:0.862068       validation_1-auc:0.84135
[149]   validation_0-auc:0.862132       validation_1-auc:0.841444
[150]   validation_0-auc:0.862236       validation_1-auc:0.841409
[151]   validation_0-auc:0.862314       validation_1-auc:0.841459
[152]   validation_0-auc:0.862584       validation_1-auc:0.841456
[153]   validation_0-auc:0.862843       validation_1-auc:0.841483
[154]   validation_0-auc:0.863033       validation_1-auc:0.841493
[155]   validation_0-auc:0.863132       validation_1-auc:0.841534
[156]   validation_0-auc:0.863423       validation_1-auc:0.841728
[157]   validation_0-auc:0.863578       validation_1-auc:0.841712
[158]   validation_0-auc:0.863872       validation_1-auc:0.841677
[159]   validation_0-auc:0.863924       validation_1-auc:0.841658
[160]   validation_0-auc:0.863985       validation_1-auc:0.841608
[161]   validation_0-auc:0.864019       validation_1-auc:0.841646
[162]   validation_0-auc:0.864049       validation_1-auc:0.841665
[163]   validation_0-auc:0.864148       validation_1-auc:0.841682
[164]   validation_0-auc:0.864221       validation_1-auc:0.841791
[165]   validation_0-auc:0.86426        validation_1-auc:0.841732
[166]   validation_0-auc:0.864309       validation_1-auc:0.841688
[167]   validation_0-auc:0.864411       validation_1-auc:0.841699
[168]   validation_0-auc:0.864581       validation_1-auc:0.841711
[169]   validation_0-auc:0.864619       validation_1-auc:0.841729
[170]   validation_0-auc:0.864709       validation_1-auc:0.841684
[171]   validation_0-auc:0.864849       validation_1-auc:0.841704
[172]   validation_0-auc:0.865047       validation_1-auc:0.841614
```

```
#XGBoost 하이퍼 파라미터 튜닝
from sklearn.model_selection import GridSearchCV

# 하이퍼 파라미터 테스트의 수행 속도를 향상시키기 위해 n_estimators를 100으로 감소
xgb_clf = XGBClassifier(n_estimators=100)

params = {'max_depth':[5, 7] , 'min_child_weight':[1,3] ,'colsample_bytree':[0.5, 0.75] }

# cv는 3으로 지정
gridcv = GridSearchCV(xgb_clf, param_grid=params, cv=3)
gridcv.fit(X_train, y_train, early_stopping_rounds=30, eval_metric="auc",
           eval_set=[(X_train, y_train), (X_test, y_test)])

print('GridSearchCV 최적 파라미터:',gridcv.best_params_)

xgb_roc_score = roc_auc_score(y_test, gridcv.predict_proba(X_test)[:,1], average='macro')
print('ROC AUC: {0:.4f}'.format(xgb_roc_score))
```

```
[44]    validation_0-auc:0.863492       validation_1-auc:0.842518
[45]    validation_0-auc:0.864029       validation_1-auc:0.842132
[46]    validation_0-auc:0.86511        validation_1-auc:0.842687
[47]    validation_0-auc:0.865765       validation_1-auc:0.84359
[48]    validation_0-auc:0.866413       validation_1-auc:0.843371
[49]    validation_0-auc:0.867119       validation_1-auc:0.843521
[50]    validation_0-auc:0.867502       validation_1-auc:0.843723
[51]    validation_0-auc:0.868477       validation_1-auc:0.844189
[52]    validation_0-auc:0.86902        validation_1-auc:0.844987
[53]    validation_0-auc:0.869566       validation_1-auc:0.845122
[54]    validation_0-auc:0.870345       validation_1-auc:0.845578
[55]    validation_0-auc:0.870851       validation_1-auc:0.844909
[56]    validation_0-auc:0.871427       validation_1-auc:0.845604
[57]    validation_0-auc:0.871999       validation_1-auc:0.845879
[58]    validation_0-auc:0.872473       validation_1-auc:0.845742
```

```
[59]    validation_0-auc:0.872719       validation_1-auc:0.845634
[60]    validation_0-auc:0.873232       validation_1-auc:0.845628
[61]    validation_0-auc:0.873589       validation_1-auc:0.84568
[62]    validation_0-auc:0.873928       validation_1-auc:0.845631
[63]    validation_0-auc:0.874383       validation_1-auc:0.845616
[64]    validation_0-auc:0.874808       validation_1-auc:0.845449
[65]    validation_0-auc:0.875236       validation_1-auc:0.845193
[66]    validation_0-auc:0.875355       validation_1-auc:0.845183
[67]    validation_0-auc:0.875632       validation_1-auc:0.845193
[68]    validation_0-auc:0.876032       validation_1-auc:0.845069
[69]    validation_0-auc:0.876196       validation_1-auc:0.845377
[70]    validation_0-auc:0.876492       validation_1-auc:0.845464
[71]    validation_0-auc:0.876668       validation_1-auc:0.845595
[72]    validation_0-auc:0.87697        validation_1-auc:0.8456
[73]    validation_0-auc:0.877193       validation_1-auc:0.845648
[74]    validation_0-auc:0.877296       validation_1-auc:0.845525
[75]    validation_0-auc:0.877818       validation_1-auc:0.845561
[76]    validation_0-auc:0.87795        validation_1-auc:0.845778
[77]    validation_0-auc:0.878264       validation_1-auc:0.845629
[78]    validation_0-auc:0.87845        validation_1-auc:0.845657
[79]    validation_0-auc:0.878938       validation_1-auc:0.845663
[80]    validation_0-auc:0.879047       validation_1-auc:0.845639
[81]    validation_0-auc:0.879656       validation_1-auc:0.845852
[82]    validation_0-auc:0.87973        validation_1-auc:0.845831
[83]    validation_0-auc:0.88004        validation_1-auc:0.84589
[84]    validation_0-auc:0.88027        validation_1-auc:0.845772
[85]    validation_0-auc:0.880452       validation_1-auc:0.845903
[86]    validation_0-auc:0.880814       validation_1-auc:0.845886
[87]    validation_0-auc:0.881284       validation_1-auc:0.846018
[88]    validation_0-auc:0.881384       validation_1-auc:0.84606
[89]    validation_0-auc:0.88152        validation_1-auc:0.846035
[90]    validation_0-auc:0.881853       validation_1-auc:0.845866
[91]    validation_0-auc:0.882031       validation_1-auc:0.845859
[92]    validation_0-auc:0.882109       validation_1-auc:0.845845
[93]    validation_0-auc:0.882179       validation_1-auc:0.845926
[94]    validation_0-auc:0.882323       validation_1-auc:0.845717
[95]    validation_0-auc:0.882451       validation_1-auc:0.845581
[96]    validation_0-auc:0.882524       validation_1-auc:0.845596
[97]    validation_0-auc:0.882802       validation_1-auc:0.845507
[98]    validation_0-auc:0.883136       validation_1-auc:0.845464
[99]    validation_0-auc:0.883211       validation_1-auc:0.845508
GridSearchCV 최적 파라미터: {'colsample_bytree': 0.5, 'max_depth': 5, 'min_child_weight':
ROC AUC: 0.8461
```

```
# n_estimators는 1000으로 증가시키고, learning_rate=0.02로 감소, reg_alpha=0.03으로 추가함
xgb_clf = XGBClassifier(n_estimators=1000, random_state=156, learning_rate=0.02, max_depth=7,₩
                        min_child_weight=1, colsample_bytree=0.75, reg_alpha=0.03)

# evaluation metric을 auc로, early stopping은 200 으로 설정하고 학습 수행
xgb_clf.fit(X_train, y_train, early_stopping_rounds=200,
            eval_metric="auc",eval_set=[(X_train, y_train), (X_test, y_test)])

xgb_roc_score = roc_auc_score(y_test, xgb_clf.predict_proba(X_test)[:,1],average='macro')
print('ROC AUC: {0:.4f}'.format(xgb_roc_score))
```

```
[352]   validation_0-auc:0.909199       validation_1-auc:0.843995
[353]   validation_0-auc:0.909223       validation_1-auc:0.843997
[354]   validation_0-auc:0.909251       validation_1-auc:0.844005
[355]   validation_0-auc:0.909347       validation_1-auc:0.844004
```

```
[356]   validation_0-auc:0.909402       validation_1-auc:0.843948
[357]   validation_0-auc:0.909506       validation_1-auc:0.843952
[358]   validation_0-auc:0.9096 validation_1-auc:0.843939
[359]   validation_0-auc:0.909709       validation_1-auc:0.84387
[360]   validation_0-auc:0.909764       validation_1-auc:0.843795
[361]   validation_0-auc:0.909851       validation_1-auc:0.843817
[362]   validation_0-auc:0.909881       validation_1-auc:0.843844
[363]   validation_0-auc:0.909925       validation_1-auc:0.843832
[364]   validation_0-auc:0.909989       validation_1-auc:0.843858
[365]   validation_0-auc:0.910067       validation_1-auc:0.843841
[366]   validation_0-auc:0.910117       validation_1-auc:0.843801
[367]   validation_0-auc:0.910138       validation_1-auc:0.84384
[368]   validation_0-auc:0.910241       validation_1-auc:0.843781
[369]   validation_0-auc:0.910315       validation_1-auc:0.843819
[370]   validation_0-auc:0.910332       validation_1-auc:0.843831
[371]   validation_0-auc:0.910433       validation_1-auc:0.84385
[372]   validation_0-auc:0.910483       validation_1-auc:0.84382
[373]   validation_0-auc:0.910544       validation_1-auc:0.843829
[374]   validation_0-auc:0.910583       validation_1-auc:0.843808
[375]   validation_0-auc:0.910673       validation_1-auc:0.843824
[376]   validation_0-auc:0.910682       validation_1-auc:0.843822
[377]   validation_0-auc:0.910775       validation_1-auc:0.843856
[378]   validation_0-auc:0.910831       validation_1-auc:0.843807
[379]   validation_0-auc:0.910856       validation_1-auc:0.843767
[380]   validation_0-auc:0.910872       validation_1-auc:0.843787
[381]   validation_0-auc:0.911075       validation_1-auc:0.843814
[382]   validation_0-auc:0.911095       validation_1-auc:0.843799
[383]   validation_0-auc:0.911174       validation_1-auc:0.843861
[384]   validation_0-auc:0.911277       validation_1-auc:0.843857
[385]   validation_0-auc:0.911381       validation_1-auc:0.843835
[386]   validation_0-auc:0.911433       validation_1-auc:0.843816
[387]   validation_0-auc:0.911502       validation_1-auc:0.843805
[388]   validation_0-auc:0.911583       validation_1-auc:0.843763
[389]   validation_0-auc:0.911676       validation_1-auc:0.843773
[390]   validation_0-auc:0.911732       validation_1-auc:0.843754
[391]   validation_0-auc:0.911744       validation_1-auc:0.843771
[392]   validation_0-auc:0.911848       validation_1-auc:0.843811
[393]   validation_0-auc:0.911876       validation_1-auc:0.843748
[394]   validation_0-auc:0.911902       validation_1-auc:0.843742
[395]   validation_0-auc:0.911934       validation_1-auc:0.843761
[396]   validation_0-auc:0.911946       validation_1-auc:0.84376
[397]   validation_0-auc:0.911956       validation_1-auc:0.84375
[398]   validation_0-auc:0.911994       validation_1-auc:0.843734
[399]   validation_0-auc:0.912041       validation_1-auc:0.843721
[400]   validation_0-auc:0.912077       validation_1-auc:0.843694
[401]   validation_0-auc:0.912108       validation_1-auc:0.843633
[402]   validation_0-auc:0.912134       validation_1-auc:0.843652
[403]   validation_0-auc:0.912181       validation_1-auc:0.843676
[404]   validation_0-auc:0.912192       validation_1-auc:0.843661
[405]   validation_0-auc:0.912239       validation_1-auc:0.84367
Stopping. Best iteration:
[205]   validation_0-auc:0.891798       validation_1-auc:0.84558

ROC AUC: 0.8456
```

```
1 #피처 중요도 그래프
2 from xgboost import plot_importance
3 import matplotlib.pyplot as plt
```

```
%matplotlib inline

fig, ax = plt.subplots(1,1,figsize=(10,8))
plot_importance(xgb_clf, ax=ax , max_num_features=20,height=0.4)
```

XGBoost의 예측 성능을 좌우하는 가장 중요한 피처는 var38, var15 순이다

LightGBM 모델 학습과 하이퍼 파라미터 튜닝

```
from lightgbm import LGBMClassifier

lgbm_clf = LGBMClassifier(n_estimators=500)

evals = [(X_test, y_test)]
lgbm_clf.fit(X_train, y_train, early_stopping_rounds=100, eval_metric="auc", eval_set=evals,
                verbose=True)

lgbm_roc_score = roc_auc_score(y_test, lgbm_clf.predict_proba(X_test)[:,1],average='macro')
print('ROC AUC: {0:.4f}'.format(lgbm_roc_score))
```

```
[88]    valid_0's auc: 0.836989 valid_0's binary_logloss: 0.140011
[89]    valid_0's auc: 0.837035 valid_0's binary_logloss: 0.140014
[90]    valid_0's auc: 0.837007 valid_0's binary_logloss: 0.140049
[91]    valid_0's auc: 0.836832 valid_0's binary_logloss: 0.140078
[92]    valid_0's auc: 0.836979 valid_0's binary_logloss: 0.14007
[93]    valid_0's auc: 0.836875 valid_0's binary_logloss: 0.140135
```

```
[94]    valid_0's auc: 0.836843 valid_0's binary_logloss: 0.140139
[95]    valid_0's auc: 0.836938 valid_0's binary_logloss: 0.140121
[96]    valid_0's auc: 0.837312 valid_0's binary_logloss: 0.14004
[97]    valid_0's auc: 0.837229 valid_0's binary_logloss: 0.140082
[98]    valid_0's auc: 0.837361 valid_0's binary_logloss: 0.140053
[99]    valid_0's auc: 0.837365 valid_0's binary_logloss: 0.140073
[100]   valid_0's auc: 0.837229 valid_0's binary_logloss: 0.140095
[101]   valid_0's auc: 0.837124 valid_0's binary_logloss: 0.14014
[102]   valid_0's auc: 0.837385 valid_0's binary_logloss: 0.140065
[103]   valid_0's auc: 0.837954 valid_0's binary_logloss: 0.139975
[104]   valid_0's auc: 0.83767  valid_0's binary_logloss: 0.140027
[105]   valid_0's auc: 0.837743 valid_0's binary_logloss: 0.140044
[106]   valid_0's auc: 0.837839 valid_0's binary_logloss: 0.140052
[107]   valid_0's auc: 0.8377   valid_0's binary_logloss: 0.140105
[108]   valid_0's auc: 0.837582 valid_0's binary_logloss: 0.140153
[109]   valid_0's auc: 0.837439 valid_0's binary_logloss: 0.140184
[110]   valid_0's auc: 0.83731  valid_0's binary_logloss: 0.140216
[111]   valid_0's auc: 0.837193 valid_0's binary_logloss: 0.140234
[112]   valid_0's auc: 0.836993 valid_0's binary_logloss: 0.140296
[113]   valid_0's auc: 0.836994 valid_0's binary_logloss: 0.140335
[114]   valid_0's auc: 0.836887 valid_0's binary_logloss: 0.140367
[115]   valid_0's auc: 0.836742 valid_0's binary_logloss: 0.140415
[116]   valid_0's auc: 0.836448 valid_0's binary_logloss: 0.140488
[117]   valid_0's auc: 0.836571 valid_0's binary_logloss: 0.140496
[118]   valid_0's auc: 0.836701 valid_0's binary_logloss: 0.140481
[119]   valid_0's auc: 0.836717 valid_0's binary_logloss: 0.140491
[120]   valid_0's auc: 0.836673 valid_0's binary_logloss: 0.140508
[121]   valid_0's auc: 0.836644 valid_0's binary_logloss: 0.14052
[122]   valid_0's auc: 0.836649 valid_0's binary_logloss: 0.140536
[123]   valid_0's auc: 0.836457 valid_0's binary_logloss: 0.140598
[124]   valid_0's auc: 0.836254 valid_0's binary_logloss: 0.140664
[125]   valid_0's auc: 0.836198 valid_0's binary_logloss: 0.140693
[126]   valid_0's auc: 0.836429 valid_0's binary_logloss: 0.140672
[127]   valid_0's auc: 0.836282 valid_0's binary_logloss: 0.14072
[128]   valid_0's auc: 0.836152 valid_0's binary_logloss: 0.140781
[129]   valid_0's auc: 0.836156 valid_0's binary_logloss: 0.140809
[130]   valid_0's auc: 0.83605  valid_0's binary_logloss: 0.140835
[131]   valid_0's auc: 0.836033 valid_0's binary_logloss: 0.140835
[132]   valid_0's auc: 0.836014 valid_0's binary_logloss: 0.140852
[133]   valid_0's auc: 0.835977 valid_0's binary_logloss: 0.1409
[134]   valid_0's auc: 0.835695 valid_0's binary_logloss: 0.140951
[135]   valid_0's auc: 0.835689 valid_0's binary_logloss: 0.140975
[136]   valid_0's auc: 0.83554  valid_0's binary_logloss: 0.141011
[137]   valid_0's auc: 0.835146 valid_0's binary_logloss: 0.141098
[138]   valid_0's auc: 0.83503  valid_0's binary_logloss: 0.141136
[139]   valid_0's auc: 0.834826 valid_0's binary_logloss: 0.141206
[140]   valid_0's auc: 0.834576 valid_0's binary_logloss: 0.141267
[141]   valid_0's auc: 0.834265 valid_0's binary_logloss: 0.141328
[142]   valid_0's auc: 0.8342   valid_0's binary_logloss: 0.141359
Early stopping, best iteration is:
[42]    valid_0's auc: 0.839599 valid_0's binary_logloss: 0.139408
ROC AUC: 0.8396
```

```
1 from sklearn.model_selection import GridSearchCV
2
3 # 하이퍼 파라미터 테스트의 수행 속도를 향상시키기 위해 n_estimators를 100으로 감소
4 lgbm_clf = LGBMClassifier(n_estimators=200)
5
```

```
params = {'num_leaves': [32, 64 ],
          'max_depth':[128, 160],
          'min_child_samples':[60, 100],
          'subsample':[0.8, 1]}


# cv는 3으로 지정
gridcv = GridSearchCV(lgbm_clf, param_grid=params, cv=3)
gridcv.fit(X_train, y_train, early_stopping_rounds=30, eval_metric="auc",
           eval_set=[(X_train, y_train), (X_test, y_test)])

print('GridSearchCV 최적 파라미터:', gridcv.best_params_)
lgbm_roc_score = roc_auc_score(y_test, gridcv.predict_proba(X_test)[:,1], average='macro')
print('ROC AUC: {0:.4f}'.format(lgbm_roc_score))
```

```
Training until validation scores don't improve for 30 rounds.
[2]     valid_0's auc: 0.828884 valid_0's binary_logloss: 0.150957      valid_1's auc: 0.
[3]     valid_0's auc: 0.838845 valid_0's binary_logloss: 0.147117      valid_1's auc: 0.
[4]     valid_0's auc: 0.843406 valid_0's binary_logloss: 0.144114      valid_1's auc: 0.
[5]     valid_0's auc: 0.846391 valid_0's binary_logloss: 0.141629      valid_1's auc: 0.
[6]     valid_0's auc: 0.848894 valid_0's binary_logloss: 0.13957       valid_1's auc: 0.
[7]     valid_0's auc: 0.851133 valid_0's binary_logloss: 0.137847      valid_1's auc: 0.
[8]     valid_0's auc: 0.852859 valid_0's binary_logloss: 0.136394      valid_1's auc: 0.
[9]     valid_0's auc: 0.854683 valid_0's binary_logloss: 0.135137      valid_1's auc: 0.
[10]    valid_0's auc: 0.855596 valid_0's binary_logloss: 0.134048      valid_1's auc: 0.
[11]    valid_0's auc: 0.856352 valid_0's binary_logloss: 0.133075      valid_1's auc: 0.
[12]    valid_0's auc: 0.857769 valid_0's binary_logloss: 0.132232      valid_1's auc: 0.
[13]    valid_0's auc: 0.859429 valid_0's binary_logloss: 0.131427      valid_1's auc: 0.
[14]    valid_0's auc: 0.86094  valid_0's binary_logloss: 0.130658      valid_1's auc: 0.
[15]    valid_0's auc: 0.862567 valid_0's binary_logloss: 0.129955      valid_1's auc: 0.
[16]    valid_0's auc: 0.864351 valid_0's binary_logloss: 0.129293      valid_1's auc: 0.
[17]    valid_0's auc: 0.865942 valid_0's binary_logloss: 0.128724      valid_1's auc: 0.
[18]    valid_0's auc: 0.867233 valid_0's binary_logloss: 0.128187      valid_1's auc: 0.
[19]    valid_0's auc: 0.868694 valid_0's binary_logloss: 0.127637      valid_1's auc: 0.
[20]    valid_0's auc: 0.86955  valid_0's binary_logloss: 0.127154      valid_1's auc: 0.
[21]    valid_0's auc: 0.870481 valid_0's binary_logloss: 0.126741      valid_1's auc: 0.
[22]    valid_0's auc: 0.871544 valid_0's binary_logloss: 0.126309      valid_1's auc: 0.
[23]    valid_0's auc: 0.872657 valid_0's binary_logloss: 0.125895      valid_1's auc: 0.
[24]    valid_0's auc: 0.873871 valid_0's binary_logloss: 0.125508      valid_1's auc: 0.
[25]    valid_0's auc: 0.875385 valid_0's binary_logloss: 0.125076      valid_1's auc: 0.
[26]    valid_0's auc: 0.876436 valid_0's binary_logloss: 0.12471       valid_1's auc: 0.
[27]    valid_0's auc: 0.877113 valid_0's binary_logloss: 0.124381      valid_1's auc: 0.
[28]    valid_0's auc: 0.878144 valid_0's binary_logloss: 0.12406       valid_1's auc: 0.
[29]    valid_0's auc: 0.879271 valid_0's binary_logloss: 0.123721      valid_1's auc: 0.
[30]    valid_0's auc: 0.87995  valid_0's binary_logloss: 0.123447      valid_1's auc: 0.
[31]    valid_0's auc: 0.88069  valid_0's binary_logloss: 0.123148      valid_1's auc: 0.
[32]    valid_0's auc: 0.881743 valid_0's binary_logloss: 0.122823      valid_1's auc: 0.
[33]    valid_0's auc: 0.882558 valid_0's binary_logloss: 0.122542      valid_1's auc: 0.
[34]    valid_0's auc: 0.883424 valid_0's binary_logloss: 0.12228       valid_1's auc: 0.
[35]    valid_0's auc: 0.88419  valid_0's binary_logloss: 0.122011      valid_1's auc: 0.
[36]    valid_0's auc: 0.885075 valid_0's binary_logloss: 0.121697      valid_1's auc: 0.
[37]    valid_0's auc: 0.885953 valid_0's binary_logloss: 0.121404      valid_1's auc: 0.
[38]    valid_0's auc: 0.887245 valid_0's binary_logloss: 0.121133      valid_1's auc: 0.
[39]    valid_0's auc: 0.887995 valid_0's binary_logloss: 0.120851      valid_1's auc: 0.
[40]    valid_0's auc: 0.888693 valid_0's binary_logloss: 0.120601      valid_1's auc: 0.
[41]    valid_0's auc: 0.889439 valid_0's binary_logloss: 0.120343      valid_1's auc: 0.
[42]    valid_0's auc: 0.890032 valid_0's binary_logloss: 0.120105      valid_1's auc: 0.
[43]    valid_0's auc: 0.890657 valid_0's binary_logloss: 0.11989       valid_1's auc: 0.
[44]    valid_0's auc: 0.891757 valid_0's binary_logloss: 0.119626      valid_1's auc: 0.
```

```
[45]    valid_0's auc: 0.892343 valid_0's binary_logloss: 0.119426      valid_1's auc: 0.
[46]    valid_0's auc: 0.892986 valid_0's binary_logloss: 0.119211      valid_1's auc: 0.
[47]    valid_0's auc: 0.89341  valid_0's binary_logloss: 0.119028      valid_1's auc: 0.
[48]    valid_0's auc: 0.894062 valid_0's binary_logloss: 0.118789      valid_1's auc: 0.
[49]    valid_0's auc: 0.894734 valid_0's binary_logloss: 0.118543      valid_1's auc: 0.
[50]    valid_0's auc: 0.895288 valid_0's binary_logloss: 0.118352      valid_1's auc: 0.
[51]    valid_0's auc: 0.895902 valid_0's binary_logloss: 0.118145      valid_1's auc: 0.
[52]    valid_0's auc: 0.896512 valid_0's binary_logloss: 0.11792       valid_1's auc: 0.
Early stopping, best iteration is:
[22]    valid_0's auc: 0.871544 valid_0's binary_logloss: 0.126309      valid_1's auc: 0.
GridSearchCV 최적 파라미터: {'max_depth': 128, 'min_child_samples': 100, 'num_leaves': 32
ROC AUC: 0.8442
```

```
1  lgbm_clf = LGBMClassifier(n_estimators=1000, num_leaves=32, sumbsample=0.8, min_child_samples
2                            max_depth=128)
3
4  evals = [(X_test, y_test)]
5  lgbm_clf.fit(X_train, y_train, early_stopping_rounds=100, eval_metric="auc", eval_set=evals,
6                  verbose=True)
7
8  lgbm_roc_score = roc_auc_score(y_test, lgbm_clf.predict_proba(X_test)[:,1],average='macro')
9  print('ROC AUC: {0:.4f}'.format(lgbm_roc_score))
```

```
[68]    valid_0's auc: 0.841335 valid_0's binary_logloss: 0.138581
[69]    valid_0's auc: 0.841157 valid_0's binary_logloss: 0.13866
[70]    valid_0's auc: 0.841363 valid_0's binary_logloss: 0.138604
[71]    valid_0's auc: 0.841247 valid_0's binary_logloss: 0.138608
[72]    valid_0's auc: 0.841129 valid_0's binary_logloss: 0.138616
[73]    valid_0's auc: 0.841231 valid_0's binary_logloss: 0.13859
[74]    valid_0's auc: 0.841063 valid_0's binary_logloss: 0.138651
[75]    valid_0's auc: 0.841226 valid_0's binary_logloss: 0.138603
[76]    valid_0's auc: 0.841163 valid_0's binary_logloss: 0.13862
[77]    valid_0's auc: 0.841357 valid_0's binary_logloss: 0.138597
[78]    valid_0's auc: 0.840873 valid_0's binary_logloss: 0.138687
[79]    valid_0's auc: 0.840753 valid_0's binary_logloss: 0.138734
[80]    valid_0's auc: 0.840892 valid_0's binary_logloss: 0.138741
[81]    valid_0's auc: 0.841138 valid_0's binary_logloss: 0.138702
[82]    valid_0's auc: 0.841058 valid_0's binary_logloss: 0.138712
[83]    valid_0's auc: 0.84078  valid_0's binary_logloss: 0.138768
[84]    valid_0's auc: 0.84061  valid_0's binary_logloss: 0.138815
[85]    valid_0's auc: 0.840361 valid_0's binary_logloss: 0.138849
[86]    valid_0's auc: 0.840272 valid_0's binary_logloss: 0.138871
[87]    valid_0's auc: 0.840075 valid_0's binary_logloss: 0.138909
[88]    valid_0's auc: 0.840357 valid_0's binary_logloss: 0.138874
[89]    valid_0's auc: 0.840169 valid_0's binary_logloss: 0.138905
[90]    valid_0's auc: 0.840125 valid_0's binary_logloss: 0.1389
[91]    valid_0's auc: 0.839679 valid_0's binary_logloss: 0.139015
[92]    valid_0's auc: 0.83983  valid_0's binary_logloss: 0.138999
[93]    valid_0's auc: 0.839799 valid_0's binary_logloss: 0.139006
[94]    valid_0's auc: 0.839851 valid_0's binary_logloss: 0.13898
[95]    valid_0's auc: 0.840149 valid_0's binary_logloss: 0.13892
[96]    valid_0's auc: 0.840139 valid_0's binary_logloss: 0.138954
[97]    valid_0's auc: 0.840006 valid_0's binary_logloss: 0.138986
[98]    valid_0's auc: 0.839846 valid_0's binary_logloss: 0.139033
[99]    valid_0's auc: 0.839834 valid_0's binary_logloss: 0.139075
[100]   valid_0's auc: 0.839472 valid_0's binary_logloss: 0.139137
[101]   valid_0's auc: 0.8394   valid_0's binary_logloss: 0.139155
[102]   valid_0's auc: 0.839448 valid_0's binary_logloss: 0.139154
```

```
[103]   valid_0's auc: 0.839538 valid_0's binary_logloss: 0.139134
[104]   valid_0's auc: 0.839496 valid_0's binary_logloss: 0.139161
[105]   valid_0's auc: 0.839596 valid_0's binary_logloss: 0.139125
[106]   valid_0's auc: 0.839639 valid_0's binary_logloss: 0.139115
[107]   valid_0's auc: 0.839791 valid_0's binary_logloss: 0.139097
[108]   valid_0's auc: 0.839814 valid_0's binary_logloss: 0.139097
[109]   valid_0's auc: 0.839695 valid_0's binary_logloss: 0.139135
[110]   valid_0's auc: 0.839293 valid_0's binary_logloss: 0.139206
[111]   valid_0's auc: 0.839318 valid_0's binary_logloss: 0.139196
[112]   valid_0's auc: 0.839257 valid_0's binary_logloss: 0.139234
[113]   valid_0's auc: 0.839251 valid_0's binary_logloss: 0.139234
[114]   valid_0's auc: 0.839284 valid_0's binary_logloss: 0.139242
[115]   valid_0's auc: 0.839367 valid_0's binary_logloss: 0.139253
[116]   valid_0's auc: 0.83942  valid_0's binary_logloss: 0.139255
[117]   valid_0's auc: 0.839346 valid_0's binary_logloss: 0.139294
[118]   valid_0's auc: 0.839207 valid_0's binary_logloss: 0.139357
[119]   valid_0's auc: 0.839105 valid_0's binary_logloss: 0.139392
[120]   valid_0's auc: 0.839302 valid_0's binary_logloss: 0.139375
[121]   valid_0's auc: 0.839406 valid_0's binary_logloss: 0.139363
[122]   valid_0's auc: 0.83949  valid_0's binary_logloss: 0.139379
Early stopping, best iteration is:
[22]    valid_0's auc: 0.844171 valid_0's binary_logloss: 0.139253
ROC AUC: 0.8442
```

✓ 7초 오후 8:38에 완료됨