Imbalanced_Nov_notebook

February 14, 2022

[2]:
```
import pandas as pd
import numpy as np
from numpy import argmax

from datetime import date, time, timedelta
import pendulum

import matplotlib.pyplot as plt
import seaborn as sns

import category_encoders as ce
from sklearn.preprocessing import LabelEncoder, StandardScaler

from sklearn.model_selection import train_test_split, cross_val_score
from sklearn.metrics import confusion_matrix, accuracy_score, precision_score,␣
↪recall_score, f1_score, precision_recall_curve
```

[90]:
```
df = pd.read_csv("orders.csv")
```

[64]:
```
df.head()
```

[64]:

	Unnamed: 0	order_id	created_at	city	category_name	experience_date	product_id	product_name	amount	device	…
0	0	2166688	2019-10-01 00:21:37+00:00	SINGAPORE	Singapore Zoo	2019-10-01	7360	Singapore Zoo with Tram Ride	110.80	HIGH_END_MOBILE	…
1	1	2166694	2019-10-01 00:29:51+00:00	SINGAPORE	Singapore Zoo	2019-10-01	7360	Singapore Zoo with Tram Ride	110.80	HIGH_END_MOBILE	…
2	2	2166764	2019-10-01 00:55:24+00:00	DUBAI	Dubai Frame	2019-12-07	8541				
3	3	2166991	2019-10-01 02:48:55+00:00	SINGAPORE	Singapore Cable Car	2019-10-01	7372				
4	4	2167074	2019-10-01 03:27:07+00:00	SINGAPORE	Universal Studios Singapore	2019-10-02	7442				

```
2         Dubai Frame Anytime Entry Tickets   32.67  HIGH_END_MOBILE  …
3  Singapore Cable Car Sky Pass: Round Trip   66.14  HIGH_END_MOBILE  …
4      Universal Studios Singapore  Tickets  106.77          DESKTOP  …

  created_dow_sin created_dow_cos created_wom_sin created_wom_cos  \
0        0.866025             0.5        0.951057        0.309017
1        0.866025             0.5        0.951057        0.309017
2        0.866025             0.5        0.951057        0.309017
3        0.866025             0.5        0.951057        0.309017
4        0.866025             0.5        0.951057        0.309017

  experience_dom_sin  experience_dom_cos experience_dow_sin  \
0           0.201299            0.979530           0.866025
1           0.201299            0.979530           0.866025
2           0.988468            0.151428          -0.866025
3           0.201299            0.979530           0.866025
4           0.394356            0.918958           0.866025

  experience_dow_cos experience_wom_sin experience_wom_cos
0                0.5           0.951057           0.309017
1                0.5           0.951057           0.309017
2                0.5           0.587785          -0.809017
3                0.5           0.951057           0.309017
4               -0.5           0.951057           0.309017

[5 rows x 37 columns]
```

[91]:
```
df = df.drop(["created_at", "experience_date", "Unnamed: 0"], axis = 1)
df.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 76829 entries, 0 to 76828
Data columns (total 34 columns):
 #   Column                        Non-Null Count  Dtype
---  ------                        --------------  -----
 0   order_id                      76829 non-null  int64
 1   city                          76829 non-null  object
 2   category_name                 76743 non-null  object
 3   product_id                    76829 non-null  int64
 4   product_name                  76829 non-null  object
 5   amount                        76829 non-null  float64
 6   device                        76829 non-null  object
 7   payment_id                    76829 non-null  object
 8   customer_ip                   76829 non-null  object
 9   customer_id                   76829 non-null  object
 10  payment_method                76829 non-null  object
 11  payment_method_provider       76829 non-null  object
 12  payment_method_bin            76778 non-null  float64
```

```
 13  payment_method_type            76741 non-null  object
 14  payment_method_product         74731 non-null  object
 15  payment_method_card_category   74228 non-null  object
 16  payment_method_issuer_bank     73690 non-null  object
 17  payment_method_issuer_country  76730 non-null  object
 18  is_fraudulent                  76829 non-null  bool
 19  time_diff                      76829 non-null  float64
 20  created_at_hod_sin             76829 non-null  float64
 21  created_at_hod_cos             76829 non-null  float64
 22  created_dom_sin                76829 non-null  float64
 23  created_dom_cos                76829 non-null  float64
 24  created_dow_sin                76829 non-null  float64
 25  created_dow_cos                76829 non-null  float64
 26  created_wom_sin                76829 non-null  float64
 27  created_wom_cos                76829 non-null  float64
 28  experience_dom_sin             76829 non-null  float64
 29  experience_dom_cos             76829 non-null  float64
 30  experience_dow_sin             76829 non-null  float64
 31  experience_dow_cos             76829 non-null  float64
 32  experience_wom_sin             76829 non-null  float64
 33  experience_wom_cos             76829 non-null  float64
dtypes: bool(1), float64(17), int64(2), object(14)
memory usage: 19.4+ MB
```

[92]:
```
df.columns
```

[92]:
```
Index(['order_id', 'city', 'category_name', 'product_id', 'product_name',
       'amount', 'device', 'payment_id', 'customer_ip', 'customer_id',
       'payment_method', 'payment_method_provider', 'payment_method_bin',
       'payment_method_type', 'payment_method_product',
       'payment_method_card_category', 'payment_method_issuer_bank',
       'payment_method_issuer_country', 'is_fraudulent', 'time_diff',
       'created_at_hod_sin', 'created_at_hod_cos', 'created_dom_sin',
       'created_dom_cos', 'created_dow_sin', 'created_dow_cos',
       'created_wom_sin', 'created_wom_cos', 'experience_dom_sin',
       'experience_dom_cos', 'experience_dow_sin', 'experience_dow_cos',
       'experience_wom_sin', 'experience_wom_cos'],
      dtype='object')
```

[93]:
```
df.is_fraudulent.value_counts(normalize=True)
```

[93]:
```
False    0.986828
True     0.013172
Name: is_fraudulent, dtype: float64
```

[94]:
```
df = df.drop(columns=["order_id", "payment_method", "payment_id",␣
↪"product_name", "payment_method_bin"])
df.head()
```

[94]:
```
        city                category_name  product_id  amount  \
0  SINGAPORE                Singapore Zoo        7360  110.80
1  SINGAPORE                Singapore Zoo        7360  110.80
2      DUBAI                  Dubai Frame        8541   32.67
3  SINGAPORE          Singapore Cable Car        7372   66.14
4  SINGAPORE  Universal Studios Singapore        7442  106.77

            device               customer_ip               customer_id  \
0  HIGH_END_MOBILE  vzH6NNH4iwxhe+eSr1L84w==  /8+13Ig7x5JbiszC0f4AHw==
1  HIGH_END_MOBILE  vzH6NNH4iwxhe+eSr1L84w==  /8+13Ig7x5JbiszC0f4AHw==
2  HIGH_END_MOBILE  cr7Phm0QLRXcaKa/bjpTDg==  gjP75WqjRZ9ZtSkHoFd+mw==
3  HIGH_END_MOBILE  ckoTvtWdM4zKRdFyDH4YcQ==  XDZdavE7L2SLdQ1zdp5FAw==
4          DESKTOP  5H/u0Mbh8cMZxsXNDZ2QYw==  9M/VfPTvlko/2Zdtg5Pg1A==

  payment_method_provider payment_method_type     payment_method_product  …  \
0              Mastercard              Credit     World MasterCard® Card  …
1              Mastercard              Credit     World MasterCard® Card  …
2              Mastercard              Credit  Platinum MasterCard® Card  …
3              Mastercard              Credit  Titanium MasterCard® Card  …
4              Mastercard               Debit     Debit MasterCard® Card  …

  created_dow_sin created_dow_cos created_wom_sin  created_wom_cos  \
0        0.866025             0.5        0.951057         0.309017
1        0.866025             0.5        0.951057         0.309017
2        0.866025             0.5        0.951057         0.309017
3        0.866025             0.5        0.951057         0.309017
4        0.866025             0.5        0.951057         0.309017

   experience_dom_sin  experience_dom_cos  experience_dow_sin  \
0            0.201299            0.979530            0.866025
1            0.201299            0.979530            0.866025
2            0.988468            0.151428           -0.866025
3            0.201299            0.979530            0.866025
4            0.394356            0.918958            0.866025

   experience_dow_cos  experience_wom_sin  experience_wom_cos
0                 0.5            0.951057            0.309017
1                 0.5            0.951057            0.309017
2                 0.5            0.587785           -0.809017
3                 0.5            0.951057            0.309017
4                -0.5            0.951057            0.309017

[5 rows x 29 columns]
```

[95]:
```
if df.shape[0] == df.drop_duplicates().shape[0] :
    print('No duplicates Found')
else:
```

```
    duplicates = df.shape[0] - df.drop_duplicates().shape[0]
    print('{} duplicates found'.format(duplicates))
```

```
6857 duplicates found
```

[96]:
```
df.nunique()
```

[96]:
```
city                              2
category_name                    96
product_id                      353
amount                         8627
device                            5
customer_ip                   44208
customer_id                   51109
payment_method_provider           5
payment_method_type               8
payment_method_product          144
payment_method_card_category      2
payment_method_issuer_bank     2046
payment_method_issuer_country   151
is_fraudulent                     2
time_diff                       117
created_at_hod_sin               24
created_at_hod_cos               22
created_dom_sin                  31
created_dom_cos                  25
created_dow_sin                   7
created_dow_cos                   5
created_wom_sin                   6
created_wom_cos                   6
experience_dom_sin               31
experience_dom_cos               25
experience_dow_sin                7
experience_dow_cos                5
experience_wom_sin                6
experience_wom_cos                6
dtype: int64
```

[97]:
```
df = df.dropna(axis = 0, how= 'any',
               subset =␣
↪['payment_method_issuer_country','payment_method_issuer_bank','payment_method_product',␣
↪'payment_method_card_category'])
```

[98]:
```
df.is_fraudulent.value_counts(normalize=True)
```

[98]:
```
False    0.985446
True     0.014554
Name: is_fraudulent, dtype: float64
```

[99]:
```
df.isna().sum()
```

[99]:
```
city                                0
category_name                      75
product_id                          0
amount                              0
device                              0
customer_ip                         0
customer_id                         0
payment_method_provider             0
payment_method_type                 0
payment_method_product              0
payment_method_card_category        0
payment_method_issuer_bank          0
payment_method_issuer_country       0
is_fraudulent                       0
time_diff                           0
created_at_hod_sin                  0
created_at_hod_cos                  0
created_dom_sin                     0
created_dom_cos                     0
created_dow_sin                     0
created_dow_cos                     0
created_wom_sin                     0
created_wom_cos                     0
experience_dom_sin                  0
experience_dom_cos                  0
experience_dow_sin                  0
experience_dow_cos                  0
experience_wom_sin                  0
experience_wom_cos                  0
dtype: int64
```

[100]:
```
list(df[df['category_name'].isna() == True]['product_id'].drop_duplicates())
```

[100]:
```
[7364]
```

[101]:
```
df['category_name'] = df['category_name'].fillna('other')
```

[102]:
```
for column in list(df.columns) :
    if df[f"{column}"].dtypes == object :
        if len(df[f"{column}"].unique()) == len(df[f"{column}"].apply(lambda x :
 ↪ x.lower()).unique()):
            print('no string duplicates in {}'.format(column))
        else:
            print('string duplicates in {}'.format(column))
```

```
no string duplicates in city
```

```
no string duplicates in category_name
no string duplicates in device
no string duplicates in customer_ip
no string duplicates in customer_id
no string duplicates in payment_method_provider
no string duplicates in payment_method_type
no string duplicates in payment_method_product
no string duplicates in payment_method_card_category
no string duplicates in payment_method_issuer_bank
no string duplicates in payment_method_issuer_country
```

[103]:
```
X = df.drop(['is_fraudulent'], axis = 1)
y = df['is_fraudulent']
```

[104]:
```
le = LabelEncoder()
X['city'] = le.fit_transform(X['city'])
X['payment_method_card_category'] = le.
 ↪fit_transform(X['payment_method_card_category'])
y = le.fit_transform(y)
```

[174]:
```
X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.2, ␣
 ↪random_state = 0, stratify=y)
```

[175]:
```
X_train.shape
```

[175]: (55628, 28)

[176]:
```
X_test.shape
```

[176]: (13908, 28)

[177]:
```
ce_target = ce.TargetEncoder(cols = ['customer_ip',␣
 ↪'customer_id','device','category_name','product_id',
                                     ␣
 ↪'payment_method_issuer_country','payment_method_issuer_bank','payment_method_provider',
                                     ␣
 ↪'payment_method_type','payment_method_product'])
X_train = ce_target.fit_transform(X_train, y_train)
X_test = ce_target.transform(X_test)
```

[178]:
```
X_train.head()
```

[178]:

	city	category_name	product_id	amount	device	customer_ip	\
74012	0	0.000359	0.000363	4.08	0.001629	0.014561	
772	0	0.001008	0.000000	176.97	0.000000	0.003916	
36668	0	0.016409	0.000000	650.03	0.048906	0.000097	
29224	0	0.000000	0.000000	57.44	0.001629	0.014561	
15716	0	0.000000	0.000000	49.78	0.001629	0.003916	

```
       customer_id  payment_method_provider  payment_method_type  \
74012     0.014561                 0.015633             0.005531
772       0.003916                 0.015633             0.005531
36668     0.000036                 0.012961             0.005531
29224     0.014561                 0.015633             0.019733
15716     0.003916                 0.015633             0.005531

       payment_method_product  …  created_dow_sin  created_dow_cos  \
74012            5.523554e-04  …    -8.660254e-01             -0.5
772              1.765252e-02  …    -8.660254e-01              0.5
36668            0.000000e+00  …     8.660254e-01             -0.5
29224            3.705241e-15  …    -2.449294e-16              1.0
15716            5.523554e-04  …     1.224647e-16             -1.0

       created_wom_sin  created_wom_cos  experience_dom_sin  \
74012    -9.510565e-01         0.309017           -0.790776
772       5.877853e-01        -0.809017            0.651372
36668    -2.449294e-16         1.000000           -0.848644
29224     9.510565e-01         0.309017            0.937752
15716    -5.877853e-01        -0.809017            0.485302

       experience_dom_cos  experience_dow_sin  experience_dow_cos  \
74012           -0.612106       -8.660254e-01                -0.5
772             -0.758758       -8.660254e-01                 0.5
36668            0.528964        1.224647e-16                -1.0
29224            0.347305       -2.449294e-16                 1.0
15716           -0.874347       -8.660254e-01                -0.5

       experience_wom_sin  experience_wom_cos
74012       -9.510565e-01            0.309017
772          5.877853e-01           -0.809017
36668       -2.449294e-16            1.000000
29224        9.510565e-01            0.309017
15716       -5.877853e-01           -0.809017

[5 rows x 28 columns]
```

[179]:
```
X_train = X_train.values
X_test = X_test.values
```

[180]:
```
continuous_columns = ['amount','time_diff']
```

[181]:
```
numerical_features = [df.columns.get_loc(c) for c in continuous_columns if c in␣
↪df]
```

[182]:
```
numerical_features
```

[182]:
```
[3, 14]
```

[183]:
```
sc = StandardScaler() ##  The issue was it is expecting an array
X_train[:,numerical_features] = sc.fit_transform(X_train[:,numerical_features])
X_test[:,numerical_features] = sc.transform(X_test[:,numerical_features])
```

[184]:
```
print(X_train.shape)
print(y_train.shape)
```

```
(55628, 28)
(55628,)
```

[188]:
```
from sklearn.linear_model import LogisticRegression
from sklearn.model_selection import cross_validate

model = LogisticRegression(max_iter = 1000)# Restricting to max Iterations to␣
 ↪avoid convergence issues
scores = cross_validate(model, X_train, y_train, cv = 5, return_train_score=␣
 ↪True)
```

[189]:
```
results = pd.DataFrame(scores)
```

[190]:
```
results
```

[190]:

	fit_time	score_time	test_score	train_score
0	0.622996	0.002003	0.993169	0.993281
1	0.518002	0.002001	0.993439	0.993416
2	0.568133	0.001000	0.994338	0.993101
3	0.610553	0.001000	0.992090	0.993618
4	0.559009	0.000993	0.993169	0.993439

[191]:
```
results.test_score.mean()
```

[191]:
```
0.9932407919904183
```

[192]:
```
results.train_score.mean()
```

[192]:
```
0.99337114261358
```

[193]:
```
from sklearn.metrics import confusion_matrix, classification_report

model.fit(X_train, y_train)
conf_mat = confusion_matrix(y_train, model.predict(X_train))
print(conf_mat)
```

```
[[54683   135]
 [  234   576]]
```

[194]:
```
tp = float(conf_mat[1][1])
tn = float(conf_mat[0][0])
fp = float(conf_mat[0][1])
fn = float(conf_mat[1][0])

print(classification_report(y_train, model.predict(X_train)))

spec = tn/(tn+fp)
sens = tp/(tp+fp)

g_mean = (spec*sens)**(0.5)

print(f"G-mean: {g_mean}")
```

```
              precision    recall  f1-score   support

           0       1.00      1.00      1.00     54818
           1       0.81      0.71      0.76       810

    accuracy                           0.99     55628
   macro avg       0.90      0.85      0.88     55628
weighted avg       0.99      0.99      0.99     55628

G-mean: 0.8989613383285026
```

[195]:
```
## Test set performance
print(classification_report(y_test, model.predict(X_test)))
```

```
              precision    recall  f1-score   support

           0       0.99      1.00      0.99     13706
           1       0.69      0.43      0.53       202

    accuracy                           0.99     13908
   macro avg       0.84      0.71      0.76     13908
weighted avg       0.99      0.99      0.99     13908
```

[135]:
```
from imblearn.under_sampling import RandomUnderSampler

rus = RandomUnderSampler(random_state=0)
X_resampled, y_resampled = rus.fit_resample(X_train, y_train)
print(X_resampled.shape)
```

```
(1620, 28)
```

[137]:
```
print(y_resampled[y_resampled == 1].shape, y_resampled[y_resampled == 0].shape)
```

```
(810,) (810,)
```

[138]:
```
mets = ["f1", "precision", "recall", "accuracy"]
score_rus = cross_validate(model, X_resampled, y_resampled, cv = 5,␣
 ↪return_train_score = True, scoring = mets)
```

```
C:\Users\sci\anaconda3\lib\site-packages\sklearn\linear_model\_logistic.py:814:
ConvergenceWarning: lbfgs failed to converge (status=1):
STOP: TOTAL NO. of ITERATIONS REACHED LIMIT.

Increase the number of iterations (max_iter) or scale the data as shown in:
    https://scikit-learn.org/stable/modules/preprocessing.html
Please also refer to the documentation for alternative solver options:
    https://scikit-learn.org/stable/modules/linear_model.html#logistic-
regression
  n_iter_i = _check_optimize_result(
C:\Users\sci\anaconda3\lib\site-packages\sklearn\linear_model\_logistic.py:814:
ConvergenceWarning: lbfgs failed to converge (status=1):
STOP: TOTAL NO. of ITERATIONS REACHED LIMIT.

Increase the number of iterations (max_iter) or scale the data as shown in:
    https://scikit-learn.org/stable/modules/preprocessing.html
Please also refer to the documentation for alternative solver options:
    https://scikit-learn.org/stable/modules/linear_model.html#logistic-
regression
  n_iter_i = _check_optimize_result(
C:\Users\sci\anaconda3\lib\site-packages\sklearn\linear_model\_logistic.py:814:
ConvergenceWarning: lbfgs failed to converge (status=1):
STOP: TOTAL NO. of ITERATIONS REACHED LIMIT.

Increase the number of iterations (max_iter) or scale the data as shown in:
    https://scikit-learn.org/stable/modules/preprocessing.html
Please also refer to the documentation for alternative solver options:
    https://scikit-learn.org/stable/modules/linear_model.html#logistic-
regression
  n_iter_i = _check_optimize_result(
C:\Users\sci\anaconda3\lib\site-packages\sklearn\linear_model\_logistic.py:814:
ConvergenceWarning: lbfgs failed to converge (status=1):
STOP: TOTAL NO. of ITERATIONS REACHED LIMIT.

Increase the number of iterations (max_iter) or scale the data as shown in:
    https://scikit-learn.org/stable/modules/preprocessing.html
Please also refer to the documentation for alternative solver options:
    https://scikit-learn.org/stable/modules/linear_model.html#logistic-
regression
  n_iter_i = _check_optimize_result(
C:\Users\sci\anaconda3\lib\site-packages\sklearn\linear_model\_logistic.py:814:
ConvergenceWarning: lbfgs failed to converge (status=1):
STOP: TOTAL NO. of ITERATIONS REACHED LIMIT.
```

```
Increase the number of iterations (max_iter) or scale the data as shown in:
    https://scikit-learn.org/stable/modules/preprocessing.html
Please also refer to the documentation for alternative solver options:
    https://scikit-learn.org/stable/modules/linear_model.html#logistic-
regression
  n_iter_i = _check_optimize_result(
```

[139]:
```
pd.DataFrame(score_rus)
```

[139]:

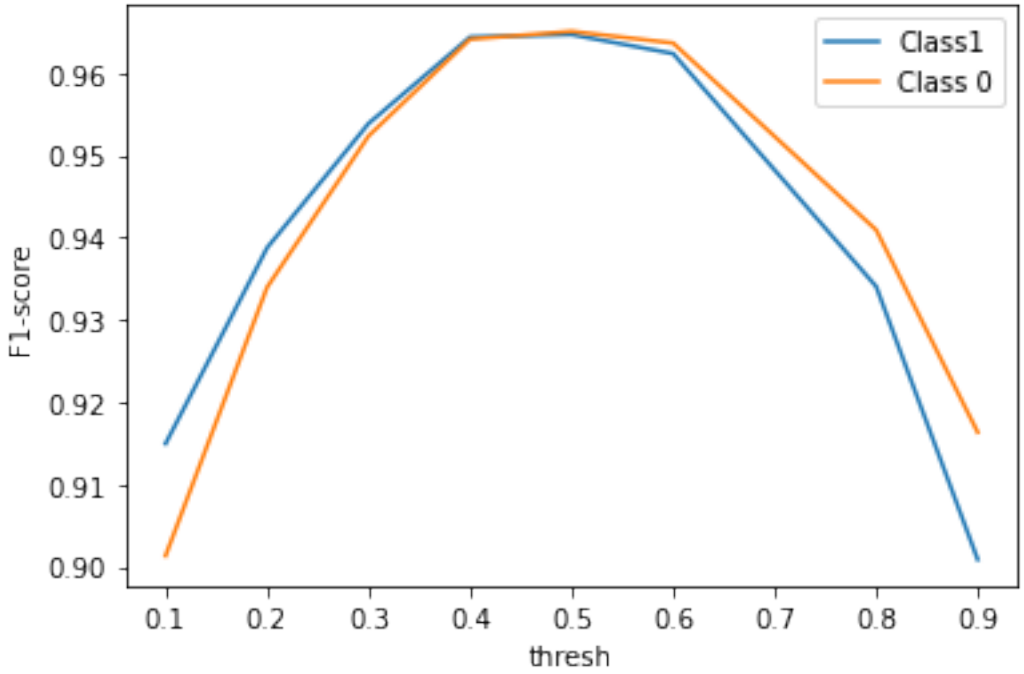
	fit_time	score_time	test_f1	train_f1	test_precision	train_precision	\
0	0.061522	0.003999	0.954407	0.962221	0.940120	0.961479	
1	0.058538	0.004002	0.947368	0.967543	0.950311	0.969040	
2	0.059002	0.004001	0.956522	0.968044	0.962500	0.977953	
3	0.060680	0.004000	0.956522	0.964981	0.962500	0.973312	
4	0.061591	0.004002	0.981132	0.963424	1.000000	0.971743	

	test_recall	train_recall	test_accuracy	train_accuracy
0	0.969136	0.962963	0.953704	0.962191
1	0.944444	0.966049	0.947531	0.967593
2	0.950617	0.958333	0.956790	0.968364
3	0.950617	0.956790	0.956790	0.965278
4	0.962963	0.955247	0.981481	0.963735

[140]:
```
model.fit(X_resampled, y_resampled)
prob = model.predict_proba(X_resampled)[:, 1]

f1_0 = []
f1_1 = []

thresh = [0.9, 0.8, 0.6, 0.5, 0.4, 0.3, 0.2, 0.1]
for i in thresh:

  preds = np.where(prob >= i, 1, 0)

  f1_0.append(f1_score(y_resampled, preds, pos_label = 0))
  f1_1.append(f1_score(y_resampled, preds))

plt.plot(thresh, f1_1)
plt.plot(thresh, f1_0)
plt.ylabel("F1-score")
plt.xlabel("thresh")
plt.legend(["Class1", "Class 0"])
plt.show()
```

```
C:\Users\sci\anaconda3\lib\site-packages\sklearn\linear_model\_logistic.py:814:
ConvergenceWarning: lbfgs failed to converge (status=1):
STOP: TOTAL NO. of ITERATIONS REACHED LIMIT.
```

```
Increase the number of iterations (max_iter) or scale the data as shown in:
    https://scikit-learn.org/stable/modules/preprocessing.html
Please also refer to the documentation for alternative solver options:
    https://scikit-learn.org/stable/modules/linear_model.html#logistic-
regression
  n_iter_i = _check_optimize_result(
```



[141]:
```
test_prob = model.predict_proba(X_test)[:, 1]
preds = np.where(test_prob>0.4, 1, 0)
print(classification_report(y_test, preds))
```

```
              precision    recall  f1-score   support

           0       1.00      0.95      0.98     13706
           1       0.23      0.93      0.36       202

    accuracy                           0.95     13908
   macro avg       0.61      0.94      0.67     13908
weighted avg       0.99      0.95      0.97     13908
```

[142]:
```
from imblearn.under_sampling import TomekLinks
from collections import Counter

tom = TomekLinks()
```

```python
X_tom, y_tom = tom.fit_resample(X_train, y_train)

print('Resampled dataset shape {}'.format(Counter(y_tom)))
```

```
Resampled dataset shape Counter({0: 54705, 1: 810})
```

[143]:
```python
score_tom = cross_validate(model, X_tom, y_tom, cv = 5, return_train_score =␣
 ↪True, scoring = mets)
pd.DataFrame(score_tom)
```

```
C:\Users\sci\anaconda3\lib\site-packages\sklearn\linear_model\_logistic.py:814:
ConvergenceWarning: lbfgs failed to converge (status=1):
STOP: TOTAL NO. of ITERATIONS REACHED LIMIT.

Increase the number of iterations (max_iter) or scale the data as shown in:
    https://scikit-learn.org/stable/modules/preprocessing.html
Please also refer to the documentation for alternative solver options:
    https://scikit-learn.org/stable/modules/linear_model.html#logistic-
regression
  n_iter_i = _check_optimize_result(
C:\Users\sci\anaconda3\lib\site-packages\sklearn\linear_model\_logistic.py:814:
ConvergenceWarning: lbfgs failed to converge (status=1):
STOP: TOTAL NO. of ITERATIONS REACHED LIMIT.

Increase the number of iterations (max_iter) or scale the data as shown in:
    https://scikit-learn.org/stable/modules/preprocessing.html
Please also refer to the documentation for alternative solver options:
    https://scikit-learn.org/stable/modules/linear_model.html#logistic-
regression
  n_iter_i = _check_optimize_result(
C:\Users\sci\anaconda3\lib\site-packages\sklearn\linear_model\_logistic.py:814:
ConvergenceWarning: lbfgs failed to converge (status=1):
STOP: TOTAL NO. of ITERATIONS REACHED LIMIT.

Increase the number of iterations (max_iter) or scale the data as shown in:
    https://scikit-learn.org/stable/modules/preprocessing.html
Please also refer to the documentation for alternative solver options:
    https://scikit-learn.org/stable/modules/linear_model.html#logistic-
regression
  n_iter_i = _check_optimize_result(
C:\Users\sci\anaconda3\lib\site-packages\sklearn\linear_model\_logistic.py:814:
ConvergenceWarning: lbfgs failed to converge (status=1):
STOP: TOTAL NO. of ITERATIONS REACHED LIMIT.

Increase the number of iterations (max_iter) or scale the data as shown in:
    https://scikit-learn.org/stable/modules/preprocessing.html
Please also refer to the documentation for alternative solver options:
    https://scikit-learn.org/stable/modules/linear_model.html#logistic-
```

```
regression
  n_iter_i = _check_optimize_result(
C:\Users\sci\anaconda3\lib\site-packages\sklearn\linear_model\_logistic.py:814:
ConvergenceWarning: lbfgs failed to converge (status=1):
STOP: TOTAL NO. of ITERATIONS REACHED LIMIT.

Increase the number of iterations (max_iter) or scale the data as shown in:
    https://scikit-learn.org/stable/modules/preprocessing.html
Please also refer to the documentation for alternative solver options:
    https://scikit-learn.org/stable/modules/linear_model.html#logistic-
regression
  n_iter_i = _check_optimize_result(
```

[143]:

	fit_time	score_time	test_f1	train_f1	test_precision	train_precision	\
0	0.349674	0.010999	0.754839	0.769357	0.790541	0.825088	
1	0.326775	0.009986	0.718644	0.765101	0.796992	0.838235	
2	0.320925	0.010997	0.748344	0.764557	0.807143	0.843575	
3	0.325865	0.010999	0.772414	0.746193	0.875000	0.825843	
4	0.424207	0.009997	0.781145	0.747885	0.859259	0.827715	

	test_recall	train_recall	test_accuracy	train_accuracy
0	0.722222	0.720679	0.993155	0.993695
1	0.654321	0.703704	0.992525	0.993695
2	0.697531	0.699074	0.993155	0.993718
3	0.691358	0.680556	0.994056	0.993245
4	0.716049	0.682099	0.994146	0.993290

[144]:
```
model.fit(X_tom, y_tom)
print(classification_report(y_tom, model.predict(X_tom)))
```

```
              precision    recall  f1-score   support

           0       1.00      1.00      1.00     54705
           1       0.83      0.70      0.76       810

    accuracy                           0.99     55515
   macro avg       0.91      0.85      0.88     55515
weighted avg       0.99      0.99      0.99     55515


C:\Users\sci\anaconda3\lib\site-packages\sklearn\linear_model\_logistic.py:814:
ConvergenceWarning: lbfgs failed to converge (status=1):
STOP: TOTAL NO. of ITERATIONS REACHED LIMIT.

Increase the number of iterations (max_iter) or scale the data as shown in:
    https://scikit-learn.org/stable/modules/preprocessing.html
Please also refer to the documentation for alternative solver options:
    https://scikit-learn.org/stable/modules/linear_model.html#logistic-
regression
```

```
  n_iter_i = _check_optimize_result(
```

[145]:
```
print(classification_report(y_test, model.predict(X_test)))
```

```
              precision    recall  f1-score   support

           0       0.99      1.00      0.99     13706
           1       0.71      0.48      0.57       202

    accuracy                           0.99     13908
   macro avg       0.85      0.74      0.78     13908
weighted avg       0.99      0.99      0.99     13908
```

[146]:
```
from imblearn.over_sampling import SMOTE

smt = SMOTE()
X_sm, y_sm = smt.fit_resample(X_train, y_train)

print('Resampled dataset shape {}'.format(Counter(y_sm)))
```

```
Resampled dataset shape Counter({0: 54818, 1: 54818})
```

[147]:
```
score_sm = cross_validate(model, X_sm, y_sm, cv = 5, return_train_score = True,␣
 ↪scoring = mets)
pd.DataFrame(score_sm)
```

```
C:\Users\sci\anaconda3\lib\site-packages\sklearn\linear_model\_logistic.py:814:
ConvergenceWarning: lbfgs failed to converge (status=1):
STOP: TOTAL NO. of ITERATIONS REACHED LIMIT.

Increase the number of iterations (max_iter) or scale the data as shown in:
    https://scikit-learn.org/stable/modules/preprocessing.html
Please also refer to the documentation for alternative solver options:
    https://scikit-learn.org/stable/modules/linear_model.html#logistic-
regression
  n_iter_i = _check_optimize_result(
C:\Users\sci\anaconda3\lib\site-packages\sklearn\linear_model\_logistic.py:814:
ConvergenceWarning: lbfgs failed to converge (status=1):
STOP: TOTAL NO. of ITERATIONS REACHED LIMIT.

Increase the number of iterations (max_iter) or scale the data as shown in:
    https://scikit-learn.org/stable/modules/preprocessing.html
Please also refer to the documentation for alternative solver options:
    https://scikit-learn.org/stable/modules/linear_model.html#logistic-
regression
  n_iter_i = _check_optimize_result(
C:\Users\sci\anaconda3\lib\site-packages\sklearn\linear_model\_logistic.py:814:
ConvergenceWarning: lbfgs failed to converge (status=1):
```

```
STOP: TOTAL NO. of ITERATIONS REACHED LIMIT.

Increase the number of iterations (max_iter) or scale the data as shown in:
    https://scikit-learn.org/stable/modules/preprocessing.html
Please also refer to the documentation for alternative solver options:
    https://scikit-learn.org/stable/modules/linear_model.html#logistic-
regression
  n_iter_i = _check_optimize_result(
C:\Users\sci\anaconda3\lib\site-packages\sklearn\linear_model\_logistic.py:814:
ConvergenceWarning: lbfgs failed to converge (status=1):
STOP: TOTAL NO. of ITERATIONS REACHED LIMIT.

Increase the number of iterations (max_iter) or scale the data as shown in:
    https://scikit-learn.org/stable/modules/preprocessing.html
Please also refer to the documentation for alternative solver options:
    https://scikit-learn.org/stable/modules/linear_model.html#logistic-
regression
  n_iter_i = _check_optimize_result(
C:\Users\sci\anaconda3\lib\site-packages\sklearn\linear_model\_logistic.py:814:
ConvergenceWarning: lbfgs failed to converge (status=1):
STOP: TOTAL NO. of ITERATIONS REACHED LIMIT.

Increase the number of iterations (max_iter) or scale the data as shown in:
    https://scikit-learn.org/stable/modules/preprocessing.html
Please also refer to the documentation for alternative solver options:
    https://scikit-learn.org/stable/modules/linear_model.html#logistic-
regression
  n_iter_i = _check_optimize_result(
```

[147]:

	fit_time	score_time	test_f1	train_f1	test_precision	train_precision	\
0	0.656133	0.017996	0.963469	0.965233	0.968396	0.968848	
1	0.773557	0.020001	0.973717	0.974952	0.979332	0.979405	
2	0.760311	0.017999	0.969825	0.970276	0.976688	0.975566	
3	0.767751	0.019963	0.972310	0.971327	0.972620	0.972437	
4	0.927605	0.020999	0.975299	0.973855	0.974677	0.974344	

	test_recall	train_recall	test_accuracy	train_accuracy
0	0.958592	0.961645	0.963654	0.965362
1	0.968166	0.970539	0.973868	0.975065
2	0.963058	0.965044	0.970037	0.970436
3	0.971999	0.970219	0.972317	0.971360
4	0.975921	0.973366	0.975282	0.973868

[148]:
```
model.fit(X_sm, y_sm)
```

```
C:\Users\sci\anaconda3\lib\site-packages\sklearn\linear_model\_logistic.py:814:
ConvergenceWarning: lbfgs failed to converge (status=1):
STOP: TOTAL NO. of ITERATIONS REACHED LIMIT.
```

```
Increase the number of iterations (max_iter) or scale the data as shown in:
    https://scikit-learn.org/stable/modules/preprocessing.html
Please also refer to the documentation for alternative solver options:
    https://scikit-learn.org/stable/modules/linear_model.html#logistic-
regression
  n_iter_i = _check_optimize_result(
```

[148]:
```
LogisticRegression()
```

[149]:
```
print(classification_report(y_test, model.predict(X_test)))
```

```
              precision    recall  f1-score   support

           0       1.00      0.97      0.99     13706
           1       0.33      0.90      0.48       202

    accuracy                           0.97     13908
   macro avg       0.66      0.93      0.73     13908
weighted avg       0.99      0.97      0.98     13908
```

[150]:
```
from imblearn.over_sampling import ADASYN

ada = ADASYN(sampling_strategy=0.3)
X_ada, y_ada = ada.fit_resample(X_train, y_train)

print('Resampled dataset shape {}'.format(Counter(y_ada)))
```

```
Resampled dataset shape Counter({0: 54818, 1: 16520})
```

[151]:
```
score_ada = cross_validate(model, X_ada, y_ada, cv = 5, return_train_score =␣
 ↪True, scoring = mets)
pd.DataFrame(score_ada)
```

```
C:\Users\sci\anaconda3\lib\site-packages\sklearn\linear_model\_logistic.py:814:
ConvergenceWarning: lbfgs failed to converge (status=1):
STOP: TOTAL NO. of ITERATIONS REACHED LIMIT.

Increase the number of iterations (max_iter) or scale the data as shown in:
    https://scikit-learn.org/stable/modules/preprocessing.html
Please also refer to the documentation for alternative solver options:
    https://scikit-learn.org/stable/modules/linear_model.html#logistic-
regression
  n_iter_i = _check_optimize_result(
C:\Users\sci\anaconda3\lib\site-packages\sklearn\linear_model\_logistic.py:814:
ConvergenceWarning: lbfgs failed to converge (status=1):
STOP: TOTAL NO. of ITERATIONS REACHED LIMIT.
```

```
Increase the number of iterations (max_iter) or scale the data as shown in:
    https://scikit-learn.org/stable/modules/preprocessing.html
Please also refer to the documentation for alternative solver options:
    https://scikit-learn.org/stable/modules/linear_model.html#logistic-
regression
  n_iter_i = _check_optimize_result(
C:\Users\sci\anaconda3\lib\site-packages\sklearn\linear_model\_logistic.py:814:
ConvergenceWarning: lbfgs failed to converge (status=1):
STOP: TOTAL NO. of ITERATIONS REACHED LIMIT.

Increase the number of iterations (max_iter) or scale the data as shown in:
    https://scikit-learn.org/stable/modules/preprocessing.html
Please also refer to the documentation for alternative solver options:
    https://scikit-learn.org/stable/modules/linear_model.html#logistic-
regression
  n_iter_i = _check_optimize_result(
C:\Users\sci\anaconda3\lib\site-packages\sklearn\linear_model\_logistic.py:814:
ConvergenceWarning: lbfgs failed to converge (status=1):
STOP: TOTAL NO. of ITERATIONS REACHED LIMIT.

Increase the number of iterations (max_iter) or scale the data as shown in:
    https://scikit-learn.org/stable/modules/preprocessing.html
Please also refer to the documentation for alternative solver options:
    https://scikit-learn.org/stable/modules/linear_model.html#logistic-
regression
  n_iter_i = _check_optimize_result(
C:\Users\sci\anaconda3\lib\site-packages\sklearn\linear_model\_logistic.py:814:
ConvergenceWarning: lbfgs failed to converge (status=1):
STOP: TOTAL NO. of ITERATIONS REACHED LIMIT.

Increase the number of iterations (max_iter) or scale the data as shown in:
    https://scikit-learn.org/stable/modules/preprocessing.html
Please also refer to the documentation for alternative solver options:
    https://scikit-learn.org/stable/modules/linear_model.html#logistic-
regression
  n_iter_i = _check_optimize_result(
```

[151]:

	fit_time	score_time	test_f1	train_f1	test_precision	train_precision	\
0	0.451033	0.013000	0.918617	0.924161	0.941532	0.946175	
1	0.405699	0.011984	0.951282	0.929850	0.954033	0.954796	
2	0.459005	0.016001	0.898076	0.926400	0.946064	0.946475	
3	0.428010	0.013991	0.930161	0.921302	0.943631	0.936943	
4	0.421019	0.012982	0.929262	0.919939	0.942421	0.945238	

	test_recall	train_recall	test_accuracy	train_accuracy
0	0.896792	0.903148	0.963204	0.965674
1	0.948547	0.906174	0.977502	0.968337

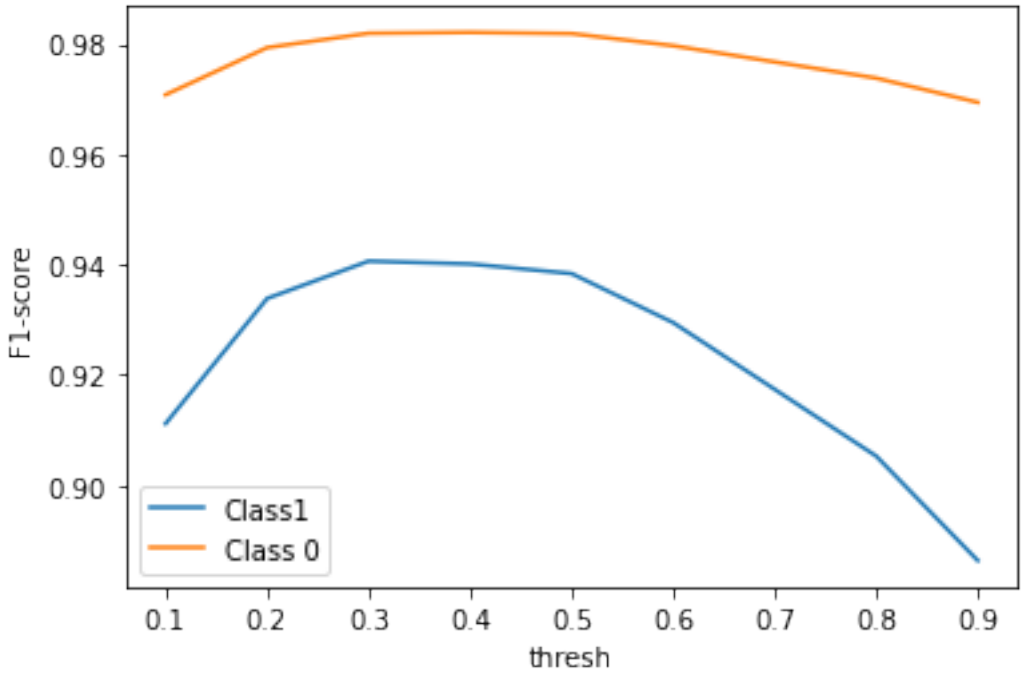
```
2     0.854722      0.907158      0.955074      0.966620
3     0.917070      0.906174      0.968108      0.964150
4     0.916465      0.895959      0.967688      0.963887
```

[152]:
```
model.fit(X_ada, y_ada)
```

```
C:\Users\sci\anaconda3\lib\site-packages\sklearn\linear_model\_logistic.py:814:
ConvergenceWarning: lbfgs failed to converge (status=1):
STOP: TOTAL NO. of ITERATIONS REACHED LIMIT.

Increase the number of iterations (max_iter) or scale the data as shown in:
    https://scikit-learn.org/stable/modules/preprocessing.html
Please also refer to the documentation for alternative solver options:
    https://scikit-learn.org/stable/modules/linear_model.html#logistic-
regression
  n_iter_i = _check_optimize_result(
```

[152]:
```
LogisticRegression()
```

[153]:
```
prob = model.predict_proba(X_ada)[:, 1]

f1_0 = []
f1_1 = []

thresh = [0.9, 0.8, 0.6, 0.5, 0.4, 0.3, 0.2, 0.1]
for i in thresh:

  preds = np.where(prob >= i, 1, 0)

  f1_0.append(f1_score(y_ada, preds, pos_label = 0))
  f1_1.append(f1_score(y_ada, preds))

plt.plot(thresh, f1_1)
plt.plot(thresh, f1_0)
plt.ylabel("F1-score")
plt.xlabel("thresh")
plt.legend(["Class1", "Class 0"])
plt.show()
```



[154]:
```
test_prob = model.predict_proba(X_test)[:, 1]
preds = np.where(test_prob>0.4, 1, 0)
print(classification_report(y_test, preds))
```

```
              precision    recall  f1-score   support

           0       1.00      0.98      0.99     13706
           1       0.44      0.90      0.59       202

    accuracy                           0.98     13908
   macro avg       0.72      0.94      0.79     13908
weighted avg       0.99      0.98      0.98     13908
```

[155]:
```
from imblearn.combine import SMOTEENN

smenn = SMOTEENN()
X_smenn, y_smenn = smenn.fit_resample(X_train, y_train)

print('Resampled dataset shape {}'.format(Counter(y_smenn)))
```

```
Resampled dataset shape Counter({1: 54055, 0: 52925})
```

[156]:
```
score_smenn = cross_validate(model, X_smenn, y_smenn, cv = 5,␣
↪return_train_score = True, scoring = mets)
```

```
pd.DataFrame(score_smenn)
```

```
C:\Users\sci\anaconda3\lib\site-packages\sklearn\linear_model\_logistic.py:814:
ConvergenceWarning: lbfgs failed to converge (status=1):
STOP: TOTAL NO. of ITERATIONS REACHED LIMIT.

Increase the number of iterations (max_iter) or scale the data as shown in:
    https://scikit-learn.org/stable/modules/preprocessing.html
Please also refer to the documentation for alternative solver options:
    https://scikit-learn.org/stable/modules/linear_model.html#logistic-
regression
  n_iter_i = _check_optimize_result(
C:\Users\sci\anaconda3\lib\site-packages\sklearn\linear_model\_logistic.py:814:
ConvergenceWarning: lbfgs failed to converge (status=1):
STOP: TOTAL NO. of ITERATIONS REACHED LIMIT.

Increase the number of iterations (max_iter) or scale the data as shown in:
    https://scikit-learn.org/stable/modules/preprocessing.html
Please also refer to the documentation for alternative solver options:
    https://scikit-learn.org/stable/modules/linear_model.html#logistic-
regression
  n_iter_i = _check_optimize_result(
C:\Users\sci\anaconda3\lib\site-packages\sklearn\linear_model\_logistic.py:814:
ConvergenceWarning: lbfgs failed to converge (status=1):
STOP: TOTAL NO. of ITERATIONS REACHED LIMIT.

Increase the number of iterations (max_iter) or scale the data as shown in:
    https://scikit-learn.org/stable/modules/preprocessing.html
Please also refer to the documentation for alternative solver options:
    https://scikit-learn.org/stable/modules/linear_model.html#logistic-
regression
  n_iter_i = _check_optimize_result(
C:\Users\sci\anaconda3\lib\site-packages\sklearn\linear_model\_logistic.py:814:
ConvergenceWarning: lbfgs failed to converge (status=1):
STOP: TOTAL NO. of ITERATIONS REACHED LIMIT.

Increase the number of iterations (max_iter) or scale the data as shown in:
    https://scikit-learn.org/stable/modules/preprocessing.html
Please also refer to the documentation for alternative solver options:
    https://scikit-learn.org/stable/modules/linear_model.html#logistic-
regression
  n_iter_i = _check_optimize_result(
C:\Users\sci\anaconda3\lib\site-packages\sklearn\linear_model\_logistic.py:814:
ConvergenceWarning: lbfgs failed to converge (status=1):
STOP: TOTAL NO. of ITERATIONS REACHED LIMIT.

Increase the number of iterations (max_iter) or scale the data as shown in:
    https://scikit-learn.org/stable/modules/preprocessing.html
```

```
Please also refer to the documentation for alternative solver options:
    https://scikit-learn.org/stable/modules/linear_model.html#logistic-
regression
  n_iter_i = _check_optimize_result(
```

[156]:

	fit_time	score_time	test_f1	train_f1	test_precision	train_precision	\
0	0.648059	0.016999	0.986049	0.986201	0.991490	0.989889	
1	0.720929	0.020001	0.974967	0.976153	0.984624	0.987529	
2	0.760998	0.017999	0.969358	0.970148	0.977611	0.976569	
3	0.767568	0.018998	0.986978	0.986581	0.992517	0.990536	
4	0.671150	0.018998	0.976783	0.976145	0.978733	0.980025	

	test_recall	train_recall	test_accuracy	train_accuracy
0	0.980668	0.982541	0.985979	0.986107
1	0.965498	0.965036	0.974949	0.976175
2	0.961243	0.963810	0.969293	0.970029
3	0.981500	0.982657	0.986913	0.986493
4	0.974840	0.972297	0.976584	0.975989

[157]:
```
model.fit(X_smenn, y_smenn)
```

```
C:\Users\sci\anaconda3\lib\site-packages\sklearn\linear_model\_logistic.py:814:
ConvergenceWarning: lbfgs failed to converge (status=1):
STOP: TOTAL NO. of ITERATIONS REACHED LIMIT.

Increase the number of iterations (max_iter) or scale the data as shown in:
    https://scikit-learn.org/stable/modules/preprocessing.html
Please also refer to the documentation for alternative solver options:
    https://scikit-learn.org/stable/modules/linear_model.html#logistic-
regression
  n_iter_i = _check_optimize_result(
```

[157]:
```
LogisticRegression()
```

[158]:
```
print(classification_report(y_test, model.predict(X_test)))
```

```
              precision    recall  f1-score   support

           0       1.00      0.97      0.98     13706
           1       0.29      0.92      0.44       202

    accuracy                           0.97     13908
   macro avg       0.65      0.94      0.71     13908
weighted avg       0.99      0.97      0.97     13908
```

[159]:
```
from imblearn.combine import SMOTETomek

smtk = SMOTETomek()
```

```
X_smtk, y_smtk = smtk.fit_resample(X_train, y_train)

print('Resampled dataset shape {}'.format(Counter(y_smtk)))
```

```
Resampled dataset shape Counter({0: 54766, 1: 54766})
```

[160]:
```
score_smtk = cross_validate(model, X_smtk, y_smtk, cv = 5, return_train_score =
 ↪True, scoring = mets)
pd.DataFrame(score_smtk)
```

```
C:\Users\sci\anaconda3\lib\site-packages\sklearn\linear_model\_logistic.py:814:
ConvergenceWarning: lbfgs failed to converge (status=1):
STOP: TOTAL NO. of ITERATIONS REACHED LIMIT.

Increase the number of iterations (max_iter) or scale the data as shown in:
    https://scikit-learn.org/stable/modules/preprocessing.html
Please also refer to the documentation for alternative solver options:
    https://scikit-learn.org/stable/modules/linear_model.html#logistic-
regression
  n_iter_i = _check_optimize_result(
C:\Users\sci\anaconda3\lib\site-packages\sklearn\linear_model\_logistic.py:814:
ConvergenceWarning: lbfgs failed to converge (status=1):
STOP: TOTAL NO. of ITERATIONS REACHED LIMIT.

Increase the number of iterations (max_iter) or scale the data as shown in:
    https://scikit-learn.org/stable/modules/preprocessing.html
Please also refer to the documentation for alternative solver options:
    https://scikit-learn.org/stable/modules/linear_model.html#logistic-
regression
  n_iter_i = _check_optimize_result(
C:\Users\sci\anaconda3\lib\site-packages\sklearn\linear_model\_logistic.py:814:
ConvergenceWarning: lbfgs failed to converge (status=1):
STOP: TOTAL NO. of ITERATIONS REACHED LIMIT.

Increase the number of iterations (max_iter) or scale the data as shown in:
    https://scikit-learn.org/stable/modules/preprocessing.html
Please also refer to the documentation for alternative solver options:
    https://scikit-learn.org/stable/modules/linear_model.html#logistic-
regression
  n_iter_i = _check_optimize_result(
C:\Users\sci\anaconda3\lib\site-packages\sklearn\linear_model\_logistic.py:814:
ConvergenceWarning: lbfgs failed to converge (status=1):
STOP: TOTAL NO. of ITERATIONS REACHED LIMIT.

Increase the number of iterations (max_iter) or scale the data as shown in:
    https://scikit-learn.org/stable/modules/preprocessing.html
Please also refer to the documentation for alternative solver options:
    https://scikit-learn.org/stable/modules/linear_model.html#logistic-
```

```
regression
  n_iter_i = _check_optimize_result(
C:\Users\sci\anaconda3\lib\site-packages\sklearn\linear_model\_logistic.py:814:
ConvergenceWarning: lbfgs failed to converge (status=1):
STOP: TOTAL NO. of ITERATIONS REACHED LIMIT.

Increase the number of iterations (max_iter) or scale the data as shown in:
    https://scikit-learn.org/stable/modules/preprocessing.html
Please also refer to the documentation for alternative solver options:
    https://scikit-learn.org/stable/modules/linear_model.html#logistic-
regression
  n_iter_i = _check_optimize_result(
```

[160]:

	fit_time	score_time	test_f1	train_f1	test_precision	train_precision	\
0	0.646359	0.016999	0.974710	0.974687	0.981843	0.981419	
1	0.650550	0.017052	0.972908	0.974079	0.978751	0.979417	
2	0.673408	0.019144	0.975015	0.975374	0.979190	0.979426	
3	0.752331	0.019999	0.965362	0.964801	0.974156	0.972850	
4	0.801777	0.018467	0.973707	0.972210	0.975357	0.975483	

	test_recall	train_recall	test_accuracy	train_accuracy
0	0.967680	0.968046	0.974894	0.974859
1	0.967135	0.968799	0.973068	0.974220
2	0.970876	0.971356	0.975121	0.975475
3	0.956724	0.956885	0.965672	0.965090
4	0.972062	0.968959	0.973751	0.972303

[161]:
```
model.fit(X_smtk, y_smtk)
```

```
C:\Users\sci\anaconda3\lib\site-packages\sklearn\linear_model\_logistic.py:814:
ConvergenceWarning: lbfgs failed to converge (status=1):
STOP: TOTAL NO. of ITERATIONS REACHED LIMIT.

Increase the number of iterations (max_iter) or scale the data as shown in:
    https://scikit-learn.org/stable/modules/preprocessing.html
Please also refer to the documentation for alternative solver options:
    https://scikit-learn.org/stable/modules/linear_model.html#logistic-
regression
  n_iter_i = _check_optimize_result(
```

[161]:
```
LogisticRegression()
```

[162]:
```
print(classification_report(y_test, model.predict(X_test)))
```

```
              precision    recall  f1-score   support

           0       1.00      0.98      0.99     13706
           1       0.36      0.91      0.52       202
```

```
    accuracy                           0.98     13908
   macro avg       0.68      0.94      0.75     13908
weighted avg       0.99      0.98      0.98     13908
```

[ ]: