In [1]:
```
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
from sklearn.metrics import roc_auc_score
from sklearn.model_selection import cross_val_score
from sklearn.ensemble import GradientBoostingClassifier
from sklearn.model_selection import train_test_split

X = np.genfromtxt('data/X.csv', delimiter=',')
Y = np.genfromtxt('data/Y.csv', delimiter=',')
```

In [2]:
```
X.shape
```

Out[2]:
```
(99990, 220)
```

In [3]:
```
Y.shape
```

Out[3]:
```
(99990,)
```

In [4]:
```
X_comp, X_test, Y_comp, Y_test = train_test_split(X, Y, test_size=0.2, rand
Xtr, Xva, Ytr, Yva = train_test_split(X_comp, Y_comp, test_size=0.2)
```

In [5]:
```
Xtr.shape
```

Out[5]:
```
(63993, 220)
```

In [6]:
```
Xva.shape
```

Out[6]:
```
(15999, 220)
```

In [20]:
```
from sklearn.metrics import classification_report
from sklearn.model_selection import GridSearchCV
import time

parameters1 = {
    "loss":["deviance"],
    "learning_rate": [0.01, 0.05, 0.075, 0.1,0.2],
    "n_estimators":[10, 100, 250, 500, 1000]
}

# 1st. Tunign n_estimators and Learning rate
GBC = GridSearchCV(GradientBoostingClassifier(), parameters1, cv=5)

print('training start')
starting_time = time.time()
GBC.fit(Xtr, Ytr)
end_time = time.time()
print("training finished, took {} seconds".format(end_time - starting_time)

# p_test3 = {'learning_rate':[0.15,0.1,0.05,0.01,0.005,0.001], 'n_estimator

# tuning = GridSearchCV(estimator=GradientBoostingClassifier(max_depth=4, m
#             param_grid = p_test3, scoring='accuracy',n_jobs=4,iid=False,
# print('training start')
# tuning.fit(Xtr,Ytr)
# print('trainnig end\n')
# tuning.grid_scores_, tuning.best_params_, tuning.best_score_
```

```
training start
training finished, took 22985.771896123886 seconds
```

In [21]:
```
gradient_boosting_classifier_roc = roc_auc_score(
   Yva, GBC.predict_proba(Xva)[:,1])
print(gradient_boosting_classifier_roc)

print("training error:", 1 - GBC.score(Xtr, Ytr))
print("validation error:", 1 - GBC.score(Xva, Yva))
```

```
0.7600362031854073
training error: 0.28381229196943414
validation error: 0.3077692355772236
```

In [24]:
```
best_estimator = GBC.best_estimator_
best_score = GBC.best_score_
best_params = GBC.best_params_
best_index = GBC.best_index_
scorer = GBC.scorer_
cv_results = GBC.cv_results_
refit_time = GBC.refit_time_
```

In [43]:
```
print('best_estimator: ', best_estimator)
print('best_score:     ', best_score)
print('best_params:    ', best_params)
```

```
best_estimator:  GradientBoostingClassifier(criterion='friedman_mse', ini
t=None,
                           learning_rate=0.1, loss='deviance', max_depth=
3,
                           max_features=None, max_leaf_nodes=None,
                           min_impurity_decrease=0.0, min_impurity_split=
None,
                           min_samples_leaf=1, min_samples_split=2,
                           min_weight_fraction_leaf=0.0, n_estimators=100
0,
                           n_iter_no_change=None, presort='auto',
                           random_state=None, subsample=1.0, tol=0.0001,
                           validation_fraction=0.1, verbose=0,
                           warm_start=False)
best_score:      0.6916850280499429
best_params:     {'learning_rate': 0.1, 'loss': 'deviance', 'n_estimator
s': 1000}
```

In [45]:
```
print('best_index:     ', best_index)
print('scorer:         ', scorer)
print('refit_time:     ', refit_time)
```

```
best_index:      19
scorer:          <function _passthrough_scorer at 0x1a1992ac20>
refit_time:      646.0830481052399
```

In [47]:
```
print('cv_results:     ', cv_results)
```

```
cv_results:      {'mean_fit_time': array([  5.32440734,  46.45433202, 11
4.95130439, 229.12073488,
       457.20627384,   5.05224881,  46.2484797 , 114.67742205,
       228.36727939, 457.45536861,   4.98603187,  45.68009925,
       114.79707303, 228.59249673, 475.68494263,   5.53827438,
        51.09562201, 126.59842916, 252.75013342, 504.0437891 ,
         5.59115834,  51.50552621, 127.61338468, 254.04380412,
       512.31585765]), 'std_fit_time': array([ 0.14453688,  0.06843351,
0.34364149,  0.34623966,  1.38477729,
        0.03113588,  0.22023396,  0.32732186,  0.67226514,  1.74548051,
        0.04586366,  0.44349906,  0.35180812,  0.96287633, 22.25149853,
        0.05797704,  0.22533611,  0.41295366,  0.84047664,  3.67097318,
        0.02447968,  0.11844933,  0.25669417,  0.49979924,  5.29967856]),
'mean_score_time': array([0.01466336, 0.02613559, 0.05179338, 0.09871078,
0.19094806,
       0.01333718, 0.02998915, 0.05780578, 0.10349646, 0.19835181,
       0.01268845, 0.03034763, 0.05813055, 0.10564132, 0.19892297,
       0.01418362, 0.03139386, 0.06057925, 0.11026602, 0.20916543,
       0.01470098, 0.03203578, 0.06288948, 0.11301103, 0.20996709]), 'std
_score_time': array([0.00059342, 0.00116663, 0.00062312, 0.00371127, 0.00
676757,
       0.00030004, 0.00089605, 0.00190857, 0.00165066, 0.00664742,
       0.00042002, 0.00107189, 0.00252892, 0.00203984, 0.00635933,
       0.00026682, 0.00138815, 0.00109997, 0.00113096, 0.00575453,
       0.00031136, 0.00059542, 0.0012066 , 0.00319272, 0.00252834]), 'par
am_learning_rate': masked_array(data=[0.01, 0.01, 0.01, 0.01, 0.01, 0.05,
0.05, 0.05, 0.05, 0.05,
                   0.05, 0.075, 0.075, 0.075, 0.075, 0.075, 0.1, 0.1, 0.
1,
                   0.1, 0.1, 0.2, 0.2, 0.2, 0.2, 0.2],
             mask=[False, False, False, False, False, False, False, Fals
e,
                   False, False, False, False, False, False, False, Fals
e,
                   False, False, False, False, False, False, False, Fals
e,
                   False],
       fill_value='?',
            dtype=object), 'param_loss': masked_array(data=['deviance',
'deviance', 'deviance', 'deviance',
                   'deviance', 'deviance', 'deviance', 'deviance',
                   'deviance', 'deviance', 'deviance', 'deviance',
                   'deviance', 'deviance', 'deviance', 'deviance',
                   'deviance', 'deviance', 'deviance', 'deviance',
                   'deviance', 'deviance', 'deviance', 'deviance',
                   'deviance'],
             mask=[False, False, False, False, False, False, False, Fals
e,
                   False, False, False, False, False, False, False, Fals
e,
                   False, False, False, False, False, False, False, Fals
e,
                   False],
       fill_value='?',
            dtype=object), 'param_n_estimators': masked_array(data=[10, 1
```

```
00, 250, 500, 1000, 10, 100, 250, 500, 1000, 10,
                   100, 250, 500, 1000, 10, 100, 250, 500, 1000, 10, 100,
                   250, 500, 1000],
             mask=[False, False, False, False, False, False, False, Fals
e,
                   False, False, False, False, False, False, False, Fals
e,
                   False, False, False, False, False, False, False, Fals
e,
                   False],
       fill_value='?',
            dtype=object), 'params': [{'learning_rate': 0.01, 'loss': 'de
viance', 'n_estimators': 10}, {'learning_rate': 0.01, 'loss': 'deviance',
'n_estimators': 100}, {'learning_rate': 0.01, 'loss': 'deviance', 'n_esti
mators': 250}, {'learning_rate': 0.01, 'loss': 'deviance', 'n_estimator
s': 500}, {'learning_rate': 0.01, 'loss': 'deviance', 'n_estimators': 100
0}, {'learning_rate': 0.05, 'loss': 'deviance', 'n_estimators': 10}, {'le
arning_rate': 0.05, 'loss': 'deviance', 'n_estimators': 100}, {'learning_
rate': 0.05, 'loss': 'deviance', 'n_estimators': 250}, {'learning_rate':
0.05, 'loss': 'deviance', 'n_estimators': 500}, {'learning_rate': 0.05,
'loss': 'deviance', 'n_estimators': 1000}, {'learning_rate': 0.075, 'los
s': 'deviance', 'n_estimators': 10}, {'learning_rate': 0.075, 'loss': 'de
viance', 'n_estimators': 100}, {'learning_rate': 0.075, 'loss': 'devianc
e', 'n_estimators': 250}, {'learning_rate': 0.075, 'loss': 'deviance', 'n
_estimators': 500}, {'learning_rate': 0.075, 'loss': 'deviance', 'n_estim
ators': 1000}, {'learning_rate': 0.1, 'loss': 'deviance', 'n_estimators':
10}, {'learning_rate': 0.1, 'loss': 'deviance', 'n_estimators': 100}, {'l
earning_rate': 0.1, 'loss': 'deviance', 'n_estimators': 250}, {'learning_
rate': 0.1, 'loss': 'deviance', 'n_estimators': 500}, {'learning_rate':
0.1, 'loss': 'deviance', 'n_estimators': 1000}, {'learning_rate': 0.2, 'l
oss': 'deviance', 'n_estimators': 10}, {'learning_rate': 0.2, 'loss': 'de
viance', 'n_estimators': 100}, {'learning_rate': 0.2, 'loss': 'deviance',
'n_estimators': 250}, {'learning_rate': 0.2, 'loss': 'deviance', 'n_estim
ators': 500}, {'learning_rate': 0.2, 'loss': 'deviance', 'n_estimators':
1000}], 'split0_test_score': array([0.56348152, 0.61348543, 0.63215876,
0.64817564, 0.66684897,
       0.60700055, 0.64762872, 0.67013048, 0.68192828, 0.69130401,
       0.60707868, 0.66020783, 0.67817798, 0.68685054, 0.69083522,
       0.61387608, 0.66856786, 0.68231893, 0.69114775, 0.69091335,
       0.63020548, 0.67927182, 0.69021017, 0.69122588, 0.69036643]), 'spl
it1_test_score': array([0.56348152, 0.60911009, 0.63465896, 0.65341042,
0.670443  ,
       0.60879756, 0.65317603, 0.67622471, 0.68896008, 0.69263224,
       0.60809438, 0.66278616, 0.68638175, 0.69255411, 0.69356981,
       0.61887647, 0.67270881, 0.68864755, 0.69310102, 0.6938042 ,
       0.6309868 , 0.68599109, 0.69255411, 0.69224158, 0.69005391]), 'spl
it2_test_score': array([0.56348152, 0.61301664, 0.63754981, 0.65247285,
0.67231815,
       0.60700055, 0.65223846, 0.67599031, 0.6857567 , 0.69294476,
       0.60700055, 0.66739589, 0.68286585, 0.69263224, 0.69372607,
       0.61278225, 0.67309946, 0.68731932, 0.69200719, 0.69349168,
       0.63278381, 0.68239706, 0.6950543 , 0.69317915, 0.6938042 ]), 'spl
it3_test_score': array([0.56348152, 0.61254786, 0.63333073, 0.64684741,
0.66598953,
       0.60684428, 0.64778498, 0.67020861, 0.68130323, 0.68692867,
       0.60723494, 0.65911399, 0.67708415, 0.68552231, 0.68864755,
       0.61262599, 0.66739589, 0.68130323, 0.68645988, 0.68849129,
```

```
       0.6307524 , 0.67864677, 0.68724119, 0.68903821, 0.68833503]), 'spl
it4_test_score': array([0.56349144, 0.61498789, 0.63608658, 0.65765414,
0.6691412 ,
       0.60951786, 0.65437212, 0.67390795, 0.68773931, 0.68875518,
       0.60951786, 0.66546847, 0.68219114, 0.68930218, 0.69094319,
       0.61514417, 0.67046964, 0.68625459, 0.69000547, 0.69172462,
       0.63225756, 0.68273814, 0.69000547, 0.69133391, 0.69070876]), 'mea
n_test_score': array([0.56348351, 0.61262951, 0.63475693, 0.65171191, 0.6
6894817,
       0.60783211, 0.65103996, 0.67329239, 0.68513744, 0.69051302,
       0.60778523, 0.66299439, 0.68134015, 0.68937228, 0.69154439,
       0.61466098, 0.67044833, 0.68516869, 0.69054428, 0.69168503,
       0.63139718, 0.68180895, 0.69101308, 0.69140375, 0.69065367]), 'std
_test_score': array([3.96833919e-06, 1.94098638e-03, 1.91638032e-03, 3.87
141947e-03,
       2.31471652e-03, 1.10746355e-03, 2.80475971e-03, 2.67449445e-03,
       3.05830342e-03, 2.32241563e-03, 9.50263253e-04, 3.11000295e-03,
       3.36405030e-03, 2.89617166e-03, 1.90360787e-03, 2.29310052e-03,
       2.23575698e-03, 2.86254769e-03, 2.28099182e-03, 1.92626559e-03,
       9.66159545e-04, 2.65092370e-03, 2.63037024e-03, 1.37743483e-03,
       1.77469503e-03]), 'rank_test_score': array([25, 22, 19, 17, 15, 2
3, 18, 13, 10,  7, 24, 16, 12,  8,  2, 21, 14,
        9,  6,  1, 20, 11,  4,  3,  5], dtype=int32)}
```

In [26]:
```
# 2nd. more estimators tests

parameters2 = {
    "loss":["deviance"],
    "learning_rate": [0.1],
    "n_estimators":[1000,1250,1500,1750]
}

GBC2 = GridSearchCV(GradientBoostingClassifier(), parameters2, cv=5)

print('training start')
starting_time = time.time()
GBC2.fit(Xtr, Ytr)
end_time = time.time()
print("training finished, took {} seconds".format(end_time - starting_time))

gradient_boosting_classifier_roc = roc_auc_score(
   Yva, GBC2.predict_proba(Xva)[:,1])
print(gradient_boosting_classifier_roc)

print("training error:", 1 - GBC2.score(Xtr, Ytr))
print("validation error:", 1 - GBC2.score(Xva, Yva))
```

```
training start
training finished, took 13299.6862180233 seconds
0.7603106412017687
training error: 0.2769834200615693
validation error: 0.30870679417463587
```

In [27]:
```
best_estimator2 = GBC2.best_estimator_
best_score2 = GBC2.best_score_
best_params2 = GBC2.best_params_
best_index2 = GBC2.best_index_
scorer2 = GBC2.scorer_
cv_results2 = GBC2.cv_results_
refit_time2 = GBC2.refit_time_
```

In [41]:
```
print('best_estimator2: ', best_estimator2)
print('best_score2:     ', best_score2)
print('best_params2:    ', best_params2)
print('best_index2:     ', best_index2)
print('scorer2:         ', scorer2)
print('cv_results2:     ', cv_results2)
print('refit_time2:     ', refit_time2)
```

```
best_estimator2:  GradientBoostingClassifier(criterion='friedman_mse', in
it=None,
                           learning_rate=0.1, loss='deviance', max_depth=
3,
                           max_features=None, max_leaf_nodes=None,
                           min_impurity_decrease=0.0, min_impurity_split=
None,
                           min_samples_leaf=1, min_samples_split=2,
                           min_weight_fraction_leaf=0.0, n_estimators=150
0,
                           n_iter_no_change=None, presort='auto',
                           random_state=None, subsample=1.0, tol=0.0001,
                           validation_fraction=0.1, verbose=0,
                           warm_start=False)
best_score2:      0.691935055396684
best_params2:     {'learning_rate': 0.1, 'loss': 'deviance', 'n_estimator
s': 1500}
best_index2:      2
scorer2:          <function _passthrough_scorer at 0x1a1992ac20>
cv_results2:      {'mean_fit_time': array([456.1479888 , 557.11392555, 67
8.12057409, 792.97217736]), 'std_fit_time': array([ 3.09444974, 14.694388
74,  3.88231629,  3.5169729 ]), 'mean_score_time': array([0.19998507, 0.2
4817305, 0.29021735, 0.34213324]), 'std_score_time': array([0.00398555,
0.0243334 , 0.01135611, 0.01119756]), 'param_learning_rate': masked_array
(data=[0.1, 0.1, 0.1, 0.1],
             mask=[False, False, False, False],
       fill_value='?',
            dtype=object), 'param_loss': masked_array(data=['deviance',
'deviance', 'deviance', 'deviance'],
             mask=[False, False, False, False],
       fill_value='?',
            dtype=object), 'param_n_estimators': masked_array(data=[1000,
1250, 1500, 1750],
             mask=[False, False, False, False],
       fill_value='?',
            dtype=object), 'params': [{'learning_rate': 0.1, 'loss': 'dev
iance', 'n_estimators': 1000}, {'learning_rate': 0.1, 'loss': 'deviance',
'n_estimators': 1250}, {'learning_rate': 0.1, 'loss': 'deviance', 'n_esti
mators': 1500}, {'learning_rate': 0.1, 'loss': 'deviance', 'n_estimator
s': 1750}], 'split0_test_score': array([0.69075709, 0.6902883 , 0.6913040
1, 0.69013204]), 'split1_test_score': array([0.6938042 , 0.69231971, 0.69
333542, 0.69247597]), 'split2_test_score': array([0.69224158, 0.69286663,
0.69411673, 0.69364794]), 'split3_test_score': array([0.68794437, 0.68849
129, 0.68966325, 0.68911634]), 'split4_test_score': array([0.69172462, 0.
69141205, 0.69125576, 0.6902399 ]), 'mean_test_score': array([0.69129436,
0.69107559, 0.69193506, 0.69112247]), 'std_test_score': array([0.0019438
8, 0.00155998, 0.00159636, 0.00167301]), 'rank_test_score': array([2, 4,
1, 3], dtype=int32)}
refit_time2:      872.3049809932709
```

In [36]:
```
# 3rd. max_depth

parameters3 = {"max_depth":[3,5,8],
               "loss":["deviance"],
               "learning_rate": [0.1],
               "n_estimators":[1500]
              }

GBC3 = GridSearchCV(GradientBoostingClassifier(), parameters3, cv=5)

print('training start')
starting_time = time.time()
GBC3.fit(Xtr, Ytr)
end_time = time.time()
print("training finished, took {} seconds".format(end_time - starting_time))

gradient_boosting_classifier_roc = roc_auc_score(
   Yva, GBC3.predict_proba(Xva)[:,1])
print(gradient_boosting_classifier_roc)

print("training error:", 1 - GBC3.score(Xtr, Ytr))
print("validation error:", 1 - GBC3.score(Xva, Yva))

best_estimator3 = GBC3.best_estimator_
best_score3 = GBC3.best_score_
best_params3 = GBC3.best_params_
best_index3 = GBC3.best_index_
scorer3 = GBC3.scorer_
cv_results3 = GBC3.cv_results_
refit_time3 = GBC3.refit_time_
```

```
training start
training finished, took 21781.42532300949 seconds
0.7603145828608906
training error: 0.27778038222930634
validation error: 0.3093943371460717
```

In [46]:
```
print('best_estimator3: ', best_estimator3)
print('best_score3:     ', best_score3)
print('best_params3:    ', best_params3)
print('best_index3:     ', best_index3)
print('scorer3:         ', scorer3)
print('cv_results3:     ', cv_results3)
print('refit_time3:     ', refit_time3)
```

```
best_estimator3:  GradientBoostingClassifier(criterion='friedman_mse', in
it=None,
                           learning_rate=0.1, loss='deviance', max_depth=
3,
                           max_features=None, max_leaf_nodes=None,
                           min_impurity_decrease=0.0, min_impurity_split=
None,
                           min_samples_leaf=1, min_samples_split=2,
                           min_weight_fraction_leaf=0.0, n_estimators=150
0,
                           n_iter_no_change=None, presort='auto',
                           random_state=None, subsample=1.0, tol=0.0001,
                           validation_fraction=0.1, verbose=0,
                           warm_start=False)
best_score3:      0.6913099870298314
best_params3:     {'learning_rate': 0.1, 'loss': 'deviance', 'max_depth':
3, 'n_estimators': 1500}
best_index3:      0
scorer3:          <function _passthrough_scorer at 0x1a1992ac20>
cv_results3:      {'mean_fit_time': array([ 678.99345841, 1312.36442642,
2188.17449841]), 'std_fit_time': array([ 3.07718779, 30.65232615, 75.8102
2355]), 'mean_score_time': array([0.29003706, 0.43334975, 0.59002728]),
'std_score_time': array([0.007324  , 0.00841676, 0.02890185]), 'param_lea
rning_rate': masked_array(data=[0.1, 0.1, 0.1],
             mask=[False, False, False],
       fill_value='?',
            dtype=object), 'param_loss': masked_array(data=['deviance',
'deviance', 'deviance'],
             mask=[False, False, False],
       fill_value='?',
            dtype=object), 'param_max_depth': masked_array(data=[3, 5,
8],
             mask=[False, False, False],
       fill_value='?',
            dtype=object), 'param_n_estimators': masked_array(data=[1500,
1500, 1500],
             mask=[False, False, False],
       fill_value='?',
            dtype=object), 'params': [{'learning_rate': 0.1, 'loss': 'dev
iance', 'max_depth': 3, 'n_estimators': 1500}, {'learning_rate': 0.1, 'lo
ss': 'deviance', 'max_depth': 5, 'n_estimators': 1500}, {'learning_rate':
0.1, 'loss': 'deviance', 'max_depth': 8, 'n_estimators': 1500}], 'split0_
test_score': array([0.69075709, 0.68810063, 0.6845066 ]), 'split1_test_sc
ore': array([0.69294476, 0.68896008, 0.68520978]), 'split2_test_score': a
rray([0.69333542, 0.68966325, 0.68466286]), 'split3_test_score': array
([0.6880225 , 0.68427221, 0.67903742]), 'split4_test_score': array([0.691
49019, 0.68961475, 0.68656716]), 'mean_test_score': array([0.69130999, 0.
68812214, 0.68399669]), 'std_test_score': array([0.00189321, 0.00200638,
```

```
0.00258368]), 'rank_test_score': array([1, 2, 3], dtype=int32)}
refit_time3:      877.0242660045624
```

In [40]:
```
# 4th. min sample split and min samples leaf

parameters4 = {
    "max_depth":[3],
    "loss":["deviance"],
    "learning_rate": [0.1],
    "n_estimators":[1500],
    "min_samples_split": [2,6,10,20,40,60],
    "min_samples_leaf": [1,3,5,7,9],}

GBC4 = GridSearchCV(GradientBoostingClassifier(), parameters4, cv=5)

print('training start')
starting_time = time.time()
GBC4.fit(Xtr, Ytr)
end_time = time.time()
print("training finished, took {} seconds".format(end_time - starting_time))

gradient_boosting_classifier_roc = roc_auc_score(
   Yva, GBC4.predict_proba(Xva)[:,1])
print(gradient_boosting_classifier_roc)

print("training error:", 1 - GBC4.score(Xtr, Ytr))
print("validation error:", 1 - GBC4.score(Xva, Yva))

best_estimator4 = GBC4.best_estimator_
best_score4 = GBC4.best_score_
best_params4 = GBC4.best_params_
best_index4 = GBC4.best_index_
scorer4 = GBC4.scorer_
cv_results4 = GBC4.cv_results_
refit_time4 = GBC4.refit_time_
```

```
--> 225                     for func, args, kwargs in self.items]
    226
    227     def __len__(self):

/opt/anaconda3/lib/python3.7/site-packages/sklearn/model_selection/_valid
ation.py in _fit_and_score(estimator, X, y, scorer, train, test, verbose,
parameters, fit_params, return_train_score, return_parameters, return_n_t
est_samples, return_times, return_estimator, error_score)
    514             estimator.fit(X_train, **fit_params)
    515         else:
--> 516             estimator.fit(X_train, y_train, **fit_params)
    517
    518     except Exception as e:

/opt/anaconda3/lib/python3.7/site-packages/sklearn/ensemble/gradient_boos
ting.py in fit(self, X, y, sample_weight, monitor)
   1544         n_stages = self._fit_stages(
   1545             X, y, raw_predictions, sample_weight, self._rng, X_va
l, y_val,
-> 1546             sample_weight_val, begin_at_stage, monitor, X_idx_sor
```

In [ ]:
```
parameters5 = {"max_features":["log2","sqrt"]}

# GBC4 = GridSearchCV(GradientBoostingClassifier(), parameters4, cv=5)

parameters6 = {"subsample":[0.5, 0.618, 0.8, 0.85, 0.9, 0.95, 1.0]}
```