

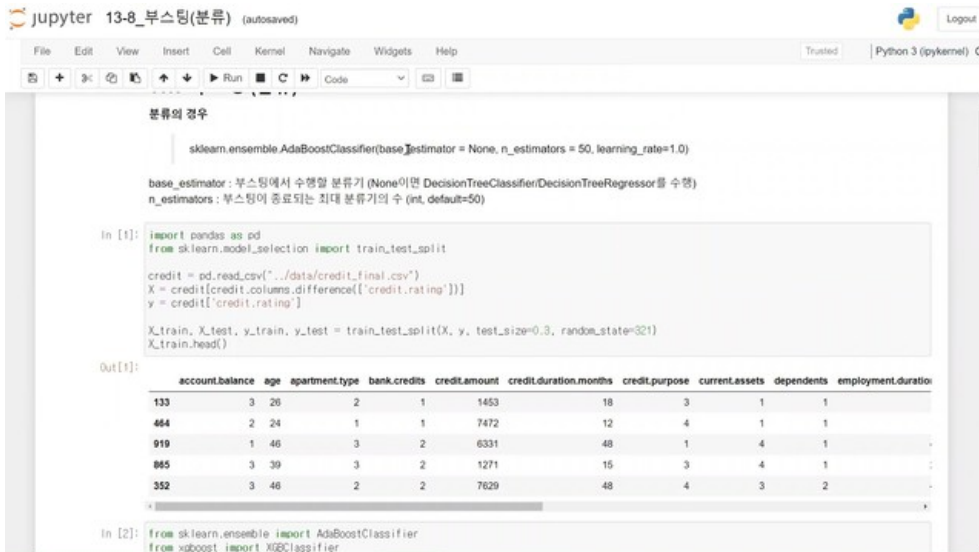
13

부스팅(분류)

[부스팅 분류]

이번 시간에는 XGBoost도 같이 실습해보겠습니다.

00:00



```

sklearn.ensemble.AdaBoostClassifier(base_estimator=None, n_estimators=50, learning_rate=1.0)

base_estimator: 부스팅에서 수행할 분류기 (None이면 DecisionTreeClassifier/DecisionTreeRegressor를 수행)
n_estimators: 부스팅이 종료되는 최대 분류기의 수 (int, default=50)

In [1]: import pandas as pd
from sklearn.model_selection import train_test_split

credit = pd.read_csv("../data/credit_final.csv")
X = credit[credit.columns.difference(['credit.rating'])]
y = credit['credit.rating']

X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.3, random_state=321)
X_train.head()

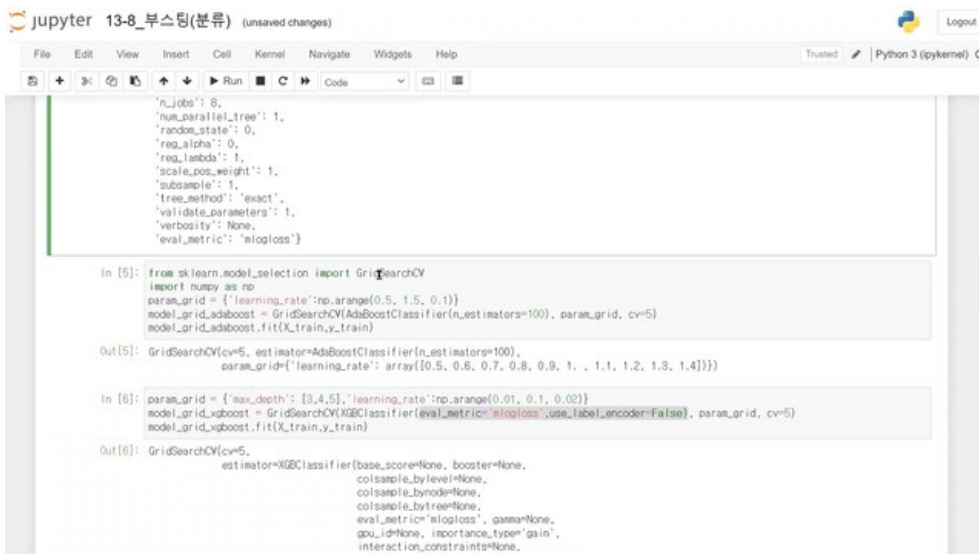
Out[1]:
  account.balance  age  apartment.type  bank.credits  credit.amount  credit.duration.months  credit.purpose  current.assets  dependents  employment.duration
133              3   26                2            1          1453                   18              3              1              1
464              2   24                1            1          7472                   12              4              1              1
919              1   46                3            2          6331                   48              1              4              1
865              3   39                3            2          1271                   15              3              4              1
352              3   46                2            2          7629                   48              4              3              2

In [2]: from sklearn.ensemble import AdaBoostClassifier
from xgboost import XGBClassifier
  
```

[그리드서치]

#그리드서치로 최적의 매개변수 값을 찾아보겠습니다.

02:45



```

'n_jobs': 8,
'num_parallel_tree': 1,
'random_state': 0,
'reg_alpha': 0,
'reg_lambda': 1,
'scale_pos_weight': 1,
'subsample': 1,
'tree_method': 'exact',
'validate_parameters': 1,
'verbosity': None,
'eval_metric': 'mlogloss')

In [5]: from sklearn.model_selection import GridSearchCV
import numpy as np
param_grid = {'learning_rate': np.arange(0.5, 1.5, 0.1)}
model_grid_adaboost = GridSearchCV(AdaBoostClassifier(n_estimators=100), param_grid, cv=5)
model_grid_adaboost.fit(X_train, y_train)

Out[5]: GridSearchCV(cv=5, estimator=AdaBoostClassifier(n_estimators=100),
  param_grid={'learning_rate': array([0.5, 0.6, 0.7, 0.8, 0.9, 1., 1.1, 1.2, 1.3, 1.4])})

In [6]: param_grid = {'max_depth': [3, 4, 5], 'learning_rate': np.arange(0.01, 0.1, 0.02)}
model_grid_xgboost = GridSearchCV(XGBClassifier(eval_metric='mlogloss', use_label_encoder=False), param_grid, cv=5)
model_grid_xgboost.fit(X_train, y_train)

Out[6]: GridSearchCV(cv=5,
  estimator=XGBClassifier(base_score=None, booster=None,
    colsample_bylevel=None,
    colsample_bynode=None,
    colsample_bytree=None,
    eval_metric='mlogloss', gamma=None,
    gpu_id=None, importance_type='gain',
    interaction_constraints=None,
    keep_features=None,
    load_parallel_interface=None,
    monotone_constraints=None,
    multi_output=False,
    num_parallel_tree=None,
    n_estimators=None,
    n_jobs=None,
    num_class=None,
    num_feature=None,
    num_leaf=None,
    num_thread=None,
    objective=None,
    random_state=None,
    reg_alpha=None,
    reg_lambda=None,
    scale_pos_weight=None,
    seed=None,
    silent=None,
    subsample=None,
    tree_method=None,
    validate_parameters=None,
    verbosity=None,
    watchlog=None)
  param_grid={'max_depth': [3, 4, 5], 'learning_rate': array([0.01, 0.02, 0.03, 0.04, 0.05, 0.06, 0.07, 0.08, 0.09, 0.1])})
  
```

[스태킹]

스택킹은 시험에 나올 확률은 적습니다.

참고만 해주시길 바랍니다^^

05:01

