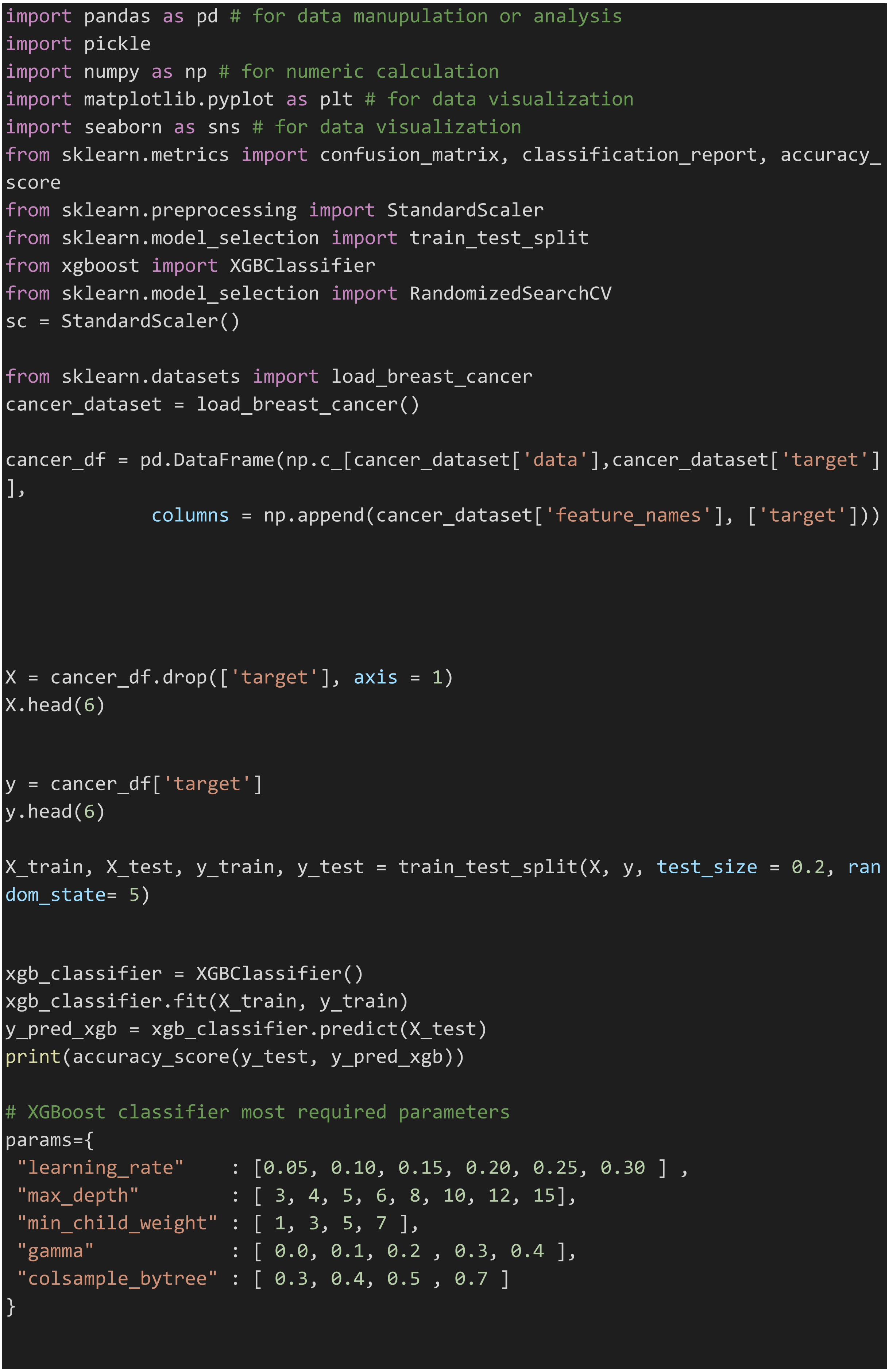

```
import pandas as pd # for data manupulation or analysis
import pickle
import numpy as np # for numeric calculation
import matplotlib.pyplot as plt # for data visualization
import seaborn as sns # for data visualization
from sklearn.metrics import confusion_matrix, classification_report, accuracy_score
from sklearn.preprocessing import StandardScaler
from sklearn.model_selection import train_test_split
from xgboost import XGBClassifier
from sklearn.model_selection import RandomizedSearchCV
sc = StandardScaler()

from sklearn.datasets import load_breast_cancer
cancer_dataset = load_breast_cancer()

cancer_df = pd.DataFrame(np.c_[cancer_dataset['data'],cancer_dataset['target']],
             columns = np.append(cancer_dataset['feature_names'], ['target']))





X = cancer_df.drop(['target'], axis = 1)
X.head(6)


y = cancer_df['target']
y.head(6)

X_train, X_test, y_train, y_test = train_test_split(X, y, test_size = 0.2, random_state= 5)


xgb_classifier = XGBClassifier()
xgb_classifier.fit(X_train, y_train)
y_pred_xgb = xgb_classifier.predict(X_test)
print(accuracy_score(y_test, y_pred_xgb))

# XGBoost classifier most required parameters
params={
 "learning_rate"    : [0.05, 0.10, 0.15, 0.20, 0.25, 0.30 ] ,
 "max_depth"        : [ 3, 4, 5, 6, 8, 10, 12, 15],
 "min_child_weight" : [ 1, 3, 5, 7 ],
 "gamma"            : [ 0.0, 0.1, 0.2 , 0.3, 0.4 ],
 "colsample_bytree" : [ 0.3, 0.4, 0.5 , 0.7 ]
}
```

```
random_search = RandomizedSearchCV(xgb_classifier, param_distributions=params,
 scoring= 'roc_auc', n_jobs= -1, verbose= 3)
print(random_search.fit(X_train, y_train))

xgb_classifier_pt = XGBClassifier(base_score=0.5, booster='gbtree', colsample_
bylevel=1,
       colsample_bynode=1, colsample_bytree=0.4, gamma=0.2,
       learning_rate=0.1, max_delta_step=0, max_depth=15,
       min_child_weight=1, missing=None, n_estimators=100, n_jobs=1,
       nthread=None, objective='binary:logistic', random_state=0,
       reg_alpha=0, reg_lambda=1, scale_pos_weight=1, seed=None,
       silent=None, subsample=1, verbosity=1)

xgb_classifier_pt.fit(X_train, y_train)
y_pred_xgb_pt = xgb_classifier_pt.predict(X_test)

print(classification_report(y_test, y_pred_xgb_pt))

# save model
pickle.dump(xgb_classifier_pt, open('breast_cancer_detector.pickle', 'wb'))

# load model
breast_cancer_detector_model = pickle.load(open('breast_cancer_detector.pickle
', 'rb'))

# predict the output
y_pred = breast_cancer_detector_model.predict(X_test)

# show the accuracy
print('Accuracy of XGBoost model = ',accuracy_score(y_test, y_pred))
```