[ML/DL] XGboost의 정의와 구현 및 hyper parameter 설정 — 나무늘보의 개발 블로그

노트북: 첫 번째 노트북
만든 날짜: 2021-01-10 오후 9:58
URL: https://continuous-development.tistory.com/191?category=736685

ML,DL

[ML/DL] XGboost의 정의와 구현 및 hyper parameter 설정

2020. 11. 13. 23:17 수정 삭제 공개

1.XGboost

1-1.xgboost란?

앙상블 모델의 한 종류인 boosting의 종류이다. 부스팅은 약한 분류기를 세트로 묶어서 정확도를 예측하는 기법이다. 또한 Xgboosting 은 gradient boosting 알고리즘의 단점을 보완해주기 위해 나왔다.
※gradient boosting 의 단점 - 느리다 , 과적합 이슈

1-2.xgboost의 특징

- gbm 보다 빠르다
- 자동 가자치기를 통해 과적합이 잘 일어나지 않는다.
- 다른 알고리즘과 연계 활용성이 좋다.
- 다양한 커스텀 최적화 옵션 제공한다. 유연성이 좋다. (ex : 조기 중단 기능)

1-3.xgboost 구현

```
# 데이터 생성 및 train test 셋 나누기
from sklearn.datasets import load_breast_cancer
dataset = load_breast_cancer()

features = dataset.data
labels = dataset.target


features = dataset.data
labels = dataset.target


cancer_df = pd.DataFrame(data=features ,  columns = dataset.feature_names)
cancer_df['target'] =  labels

X_train , X_test, y_train, y_test = train_test_split(features, labels,test_size=0.2,random_state = 100)
```

```
from xgboost import XGBClassifier

sklearn_xgboost_model = XGBClassifier(n_estimators=400,learning_rate=0.1,max_depth=3)
sklearn_xgboost_model.fit(X_train,y_train)

y_pred = sklearn_xgboost_model.predict(X_test)

def classifier_eval(y_test , y_pred) :
    print('오차행렬 : ' , confusion_matrix(y_test, y_pred))
    print('정확도   : ' , accuracy_score(y_test, y_pred))
    print('정밀도   : ' , precision_score(y_test, y_pred))
    print('재현율   : ' , recall_score(y_test, y_pred))
    print('F1       : ' , f1_score(y_test, y_pred))
    print('AUC      : ' , roc_auc_score(y_test, y_pred))


classifier_eval(y_test , y_pred)
```

```
오차행렬 :  [[46  3]
 [ 1 64]]
정확도   :  0.9649122807017544
정밀도   :  0.9552238805970149
재현율   :  0.9846153846153847
F1       :  0.9696969696969696
AUC      :  0.9616954474097332
```

2.XGboost 하이퍼 파라미터

1-1.하이퍼 파라미터의 종류

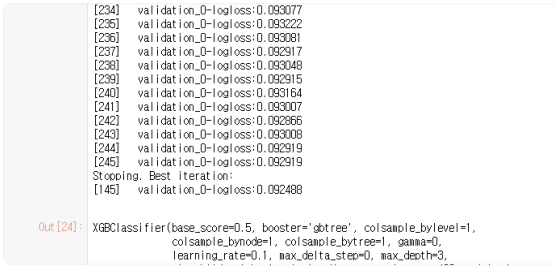
- learning_rate - 학습률 (디폴트는 0.3)
- n_estimators - 학습기의 개수(반복 수행 횟수)
- min_child_weight - leaf와 유사 , 과적합 조절용
- max_depth - 트리의 최대 깊이
- subsample - 샘플링하는 비율
- early_stopping_rounds : 더 이상 비용 평가 지표가 감소하지 않는 최대 반복 횟수(조기 중단 기능)
- eval_metric : 반복 수행 시 사용하는 비용 평가지표
- eval_set : 평가를 수행하는 별도의 검증 데이터 세트, 일반적으로 검증 데이터 세트에서 반복적으로 비용 감소 성능 평가

1-2. 하이퍼 파라미터 구현

```
sklearn_xgboost_model = XGBClassifier(n_estimators=400,learning_rate=0.1,max_depth=3)
```

```
sklearn_xgboost_model.fit(X_train,y_train,
                          early_stopping_rounds=100, # 이걸로 성능이 향상되지 않는 부분을 찾는다.
                          eval_metric='logloss',
                          eval_set = [(X_test,y_test)], # 원래는 새로운 데이터를 넣어줘야 한다. 과적합의 경우가 
                          verbose=True)
```

```
[142]   validation_0-logloss:0.092969
[143]   validation_0-logloss:0.093209
[144]   validation_0-logloss:0.092769
[145]   validation_0-logloss:0.092488
[146]   validation_0-logloss:0.092776
[147]   validation_0-logloss:0.092513
[148]   validation_0-logloss:0.09273
[149]   validation_0-logloss:0.092895
[150]   validation_0-logloss:0.09293
[151]   validation_0-logloss:0.092806
[152]   validation_0-logloss:0.092956
```



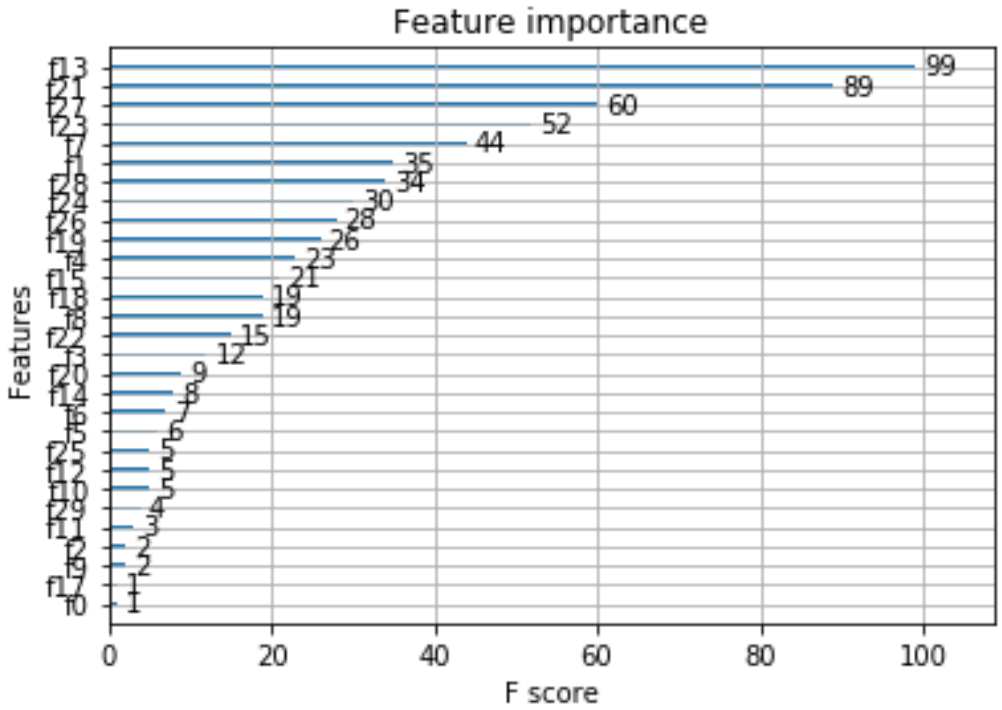
early stopping을 통해 제일 낮은 지점을 구한 다음에 그 기준으로 100번 정도 더 돌려본다.
정확도가 안 떨어지는데 무의미하게 계속 돌리기보다는 100번 정도만 더 보고 판단한다.

```
y_pred145 = sklearn_xgboost_model.predict(X_test)
classifier_eval(y_test,y_pred145)
```

```
오차행렬 :  [[47  2]
 [ 1 64]]
정확도    :  0.9736842105263158
정밀도    :  0.9696969696969697
재현율    :  0.9846153846153847
F1        :  0.9770992366412214
AUC       :  0.9718995290423862
```

```
# 피처 중요도 시각화
from xgboost import plot_importance

# fig, ax = plt.subplot(figsize=(15,5))
plot_importance(sklearn_xgboost_model)
```



'ML,DL' 카테고리의 다른 글□

[ML/DL] 회귀(Regression)의 정의와 구현□

[ML/DL] 군집화의 정의와 종류 및 구현□

[ML/DL] XGboost의 정의와 구현 및 hyper parameter 설정

[ML/DL] 앙상블 학습 (Ensemble Learning): 3.Boosting(부스팅)이란?

[ML/DL] 앙상블 학습 (Ensemble Learning): 2. Voting(보팅)이란?

[ML/DL] 앙상블 학습 (Ensemble Learning): 1. bagging(배깅)이란?

XGBoost XGboost 사용법 XGboost 정의 XGboost 하이퍼파라미터 XGboosting XGboosting 정의

나아무늘보

혼자 끄적끄적하는 블로그 입니다.