7장 앙상블 학습과 랜덤 포레스트 2부

주요 내용

- 앙상블 학습
- 배깅
    - 배깅과 페이스팅
    - 램덤포레스트
- **부스팅**
    - **그레이디언트 부스팅**
    - **XGBoost**

7.6 부스팅

- 부스팅(boosting): 성능이 약한 학습기의 여러 개를 선형으로 연결하여 강한 성능의 학습기를 만드는 앙상블 기법. 대표적 알고리즘은 다음과 같음.
    - 에이다부스트AdaBoost
    - 그레이디언트 부스팅Gradient Boosting
    - XGBoost
- 순차적으로 이전 학습기의 결과를 바탕으로 성능을 조금씩 높혀감. 즉, 편향을 줄여나감.
- 성능이 약한 예측기의 단점을 보완하여 좋은 성능의 예측기를 훈련해 나가는 것이 부스팅의 기본 아이디어
- 순차적으로 학습하기에 배깅/페이스팅에 비해 확장성이 떨어짐

그레이디언트 부스팅

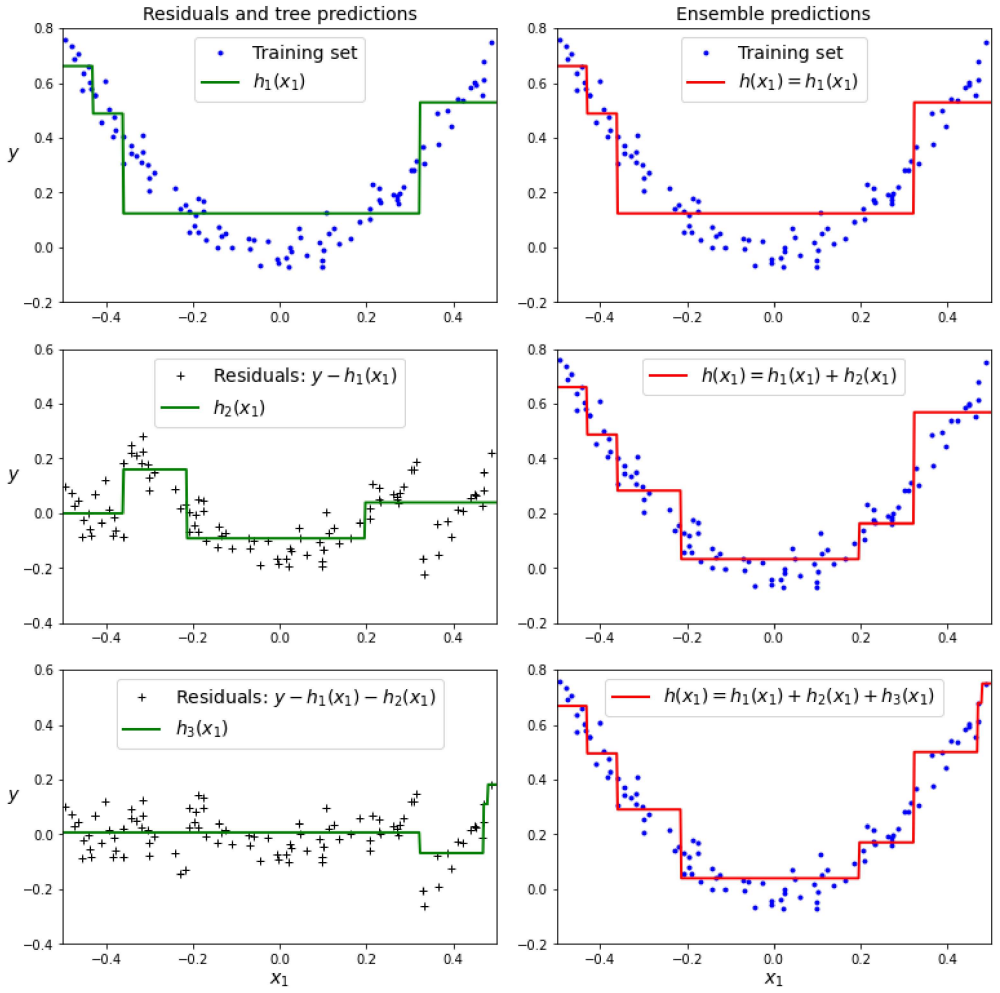
- 이전 학습기에 의한 오차를 보정하도록 새로운 예측기를 순차적으로 추가하는 아이디어는 에이다부스트와 동일
- 샘플의 가중치를 수정하는 대신 이전 예측기가 만든 **잔차**residual error에 대해 새로운 예측기를 학습시킴
- 잔차: 예측값과 실제값 사이의 오차

사이킷런 그레이디언트 부스팅 모델

- 분류 모델: `GradientBoostingClassifier`
    - `RandomForestClassifier`와 비슷한 하이퍼파라미터를 제공
- 회귀 모델: `GradientBoostingRegressor`
    - `RandomForestRegressor`와 비슷한 하이퍼파라미터를 제공

예제: 그레이디언트 부스팅 (회귀)

```
gbrt = GradientBoostingRegressor(max_depth=2, n_estimators=3,
learning_rate=1.0, random_state=42)
```

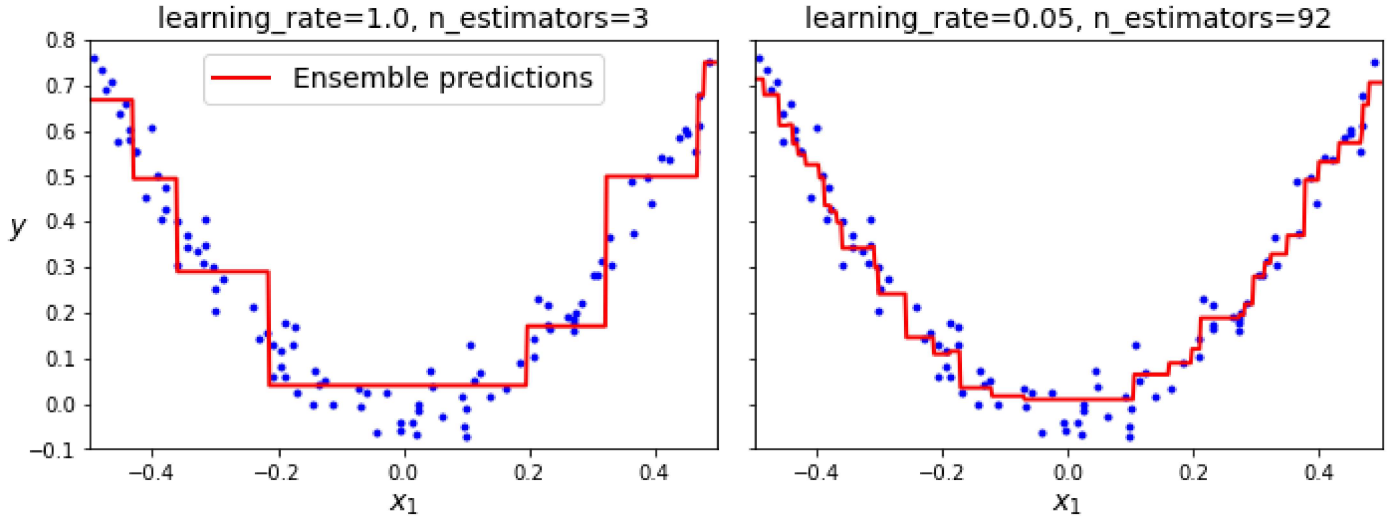


learning_rate(학습률)

- learnign_rate는 기존에 설명한 학습률과 다른 의미의 학습률.
    - 각 결정트리의 기여도 조절에 사용

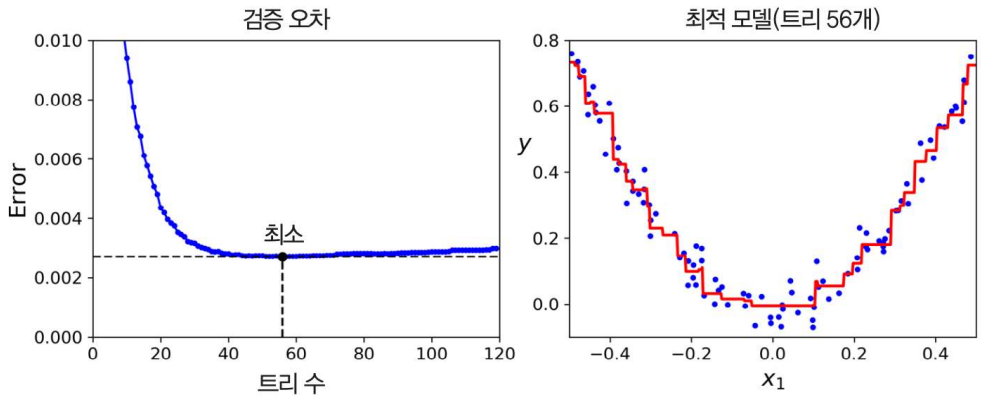
- 수축shrinkage 규제: 학습률을 낮게 정하면 많은 수의 결정트리 필요하지만 성능 좋아짐.

- 이전 결정트리에서 학습된 값을 전달할 때 사용되는 비율
  - 1.0이면 그대로 전달
  - 1.0보다 작으면 해당 비율 만큼 조금만 전달



최적의 결정트리 수 확인법

- 조기종료 기법 적용: `n_iter_no_change=None`이 기본값. 임의의 정수로 지정.



확률적 그레이디언트 부스팅

- 각 결정트리가 훈련에 사용할 훈련 샘플의 비율을 지정하여 학습: `subsample=0.25` 등 비율 지정
- 훈련 속도 빨라짐.
- 편향 높아지지만, 분산 낮아짐.

히스토그램 그레이디언트 부스팅

In [ ]:

In [ ]:

In [ ]:

In [ ]:

XGBoost

- Extreme Gradient Boosting의 줄임말.
- 빠른 속도, 확장성, 이식성 뛰어남.
- 조기종료 등 다양한 기능 제공.

```
import xgboost
  xgb_reg = xgboost.XGBRegressor(random_state=42)
  xgb_reg.fit(X_train, y_train,
              eval_set=[(X_val, y_val)],
              early_stopping_rounds=2)
```