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 학습기를 순차적으로 보다 강한 성능의 학습기로 만들어 가는 기법.
- 순차적으로 이전 학습기의 결과를 바탕으로 예측값의 정확도를 조금씩 높혀감. 즉, 편향을 줄여나감.
- 부스팅 기법을 사용하는 대표적인 모델
    - 에이다부스트AdaBoost
    - 그레이디언트 부스팅Gradient Boosting
    - XGBoost
- 여기서는 가장 성능이 좋은 그레이디언트 부스팅과 XGBoost 소개

그레이디언트 부스팅

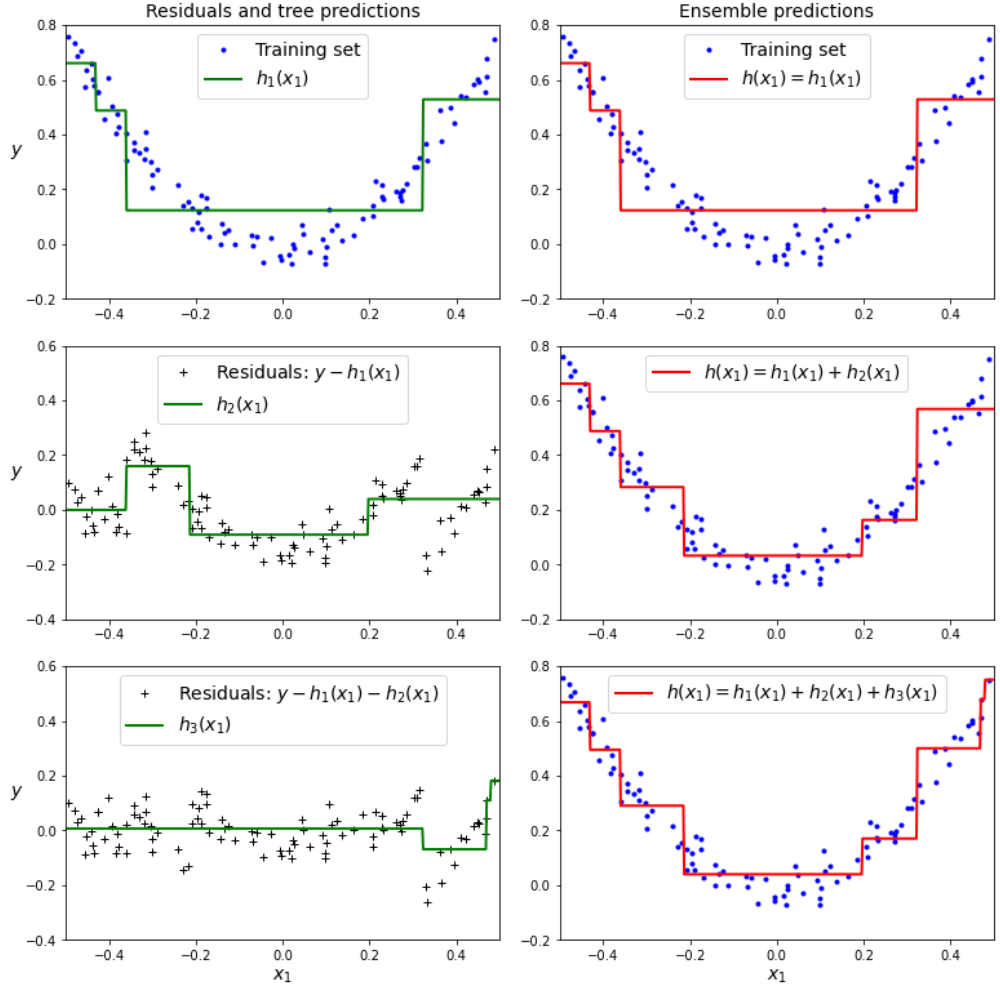
- 이전 모델에 의해 생성된 **잔차**residual error를 보정하도록 새로운 예측기 훈련
- 잔차: 예측값과 실제값 사이의 오차
- 모델은 주고 결정트리 사용

사이킷런 그레이디언트 부스팅 모델

- 결정트리 모델을 연속적으로 훈련시킴.
- 분류 모델: `GradientBoostingClassifier`
- 회귀 모델: `GradientBoostingRegressor`

예제: 그레이디언트 부스팅 (회귀)

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



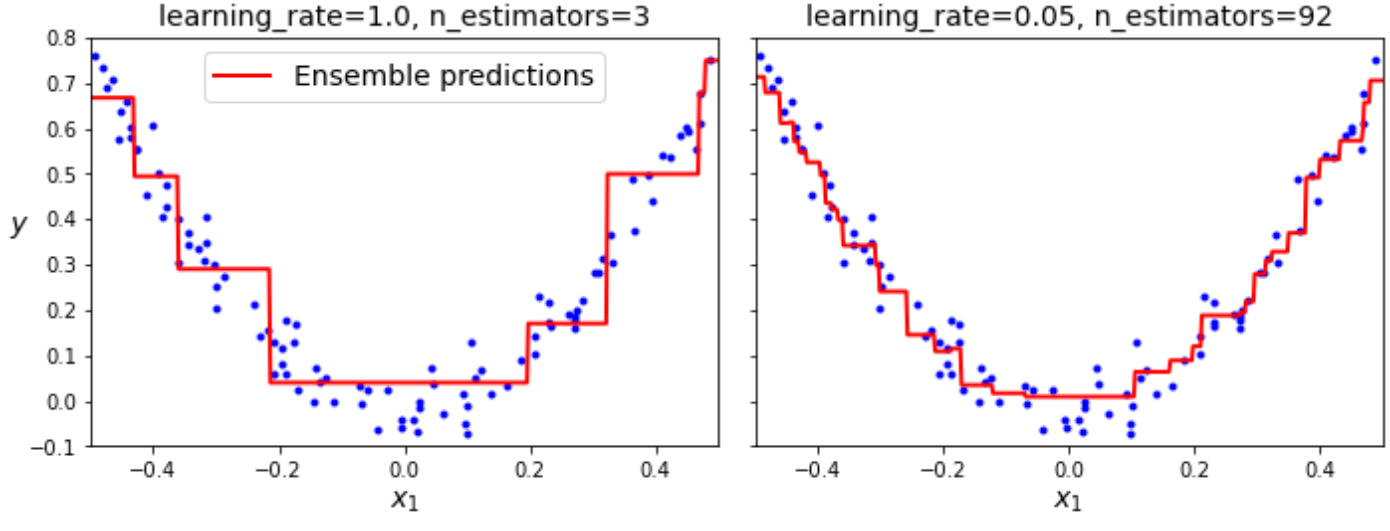
학습률과 수축 규제

- `learnign_rate`
    - 훈련된 결정 트리 모델 각각이 최종 예측값을 계산할 때의 기여도 결정
    - 경사하강법의 학습률과 다르지만 최종 모델에 수렴하는 속도를 조절한다는 차원에서 동일한 기능 수행.

- 수축shrinkage 규제
    - 훈련에 사용되는 각 모델의 기여도를 줄이는 방식으로 훈련 규제
    - 학습률을 낮게 정하면 많은 수의 결정트리 필요하지만 성능은 일반적으로 좋아짐.

예제: 수축 규제

- 아래 왼쪽: 학습률=1. 3 개의 결정트리 학습. 과소적합.
- 아래 오른쪽: 학습률=0.05. 92개의 결정트리 학습. 적절한 모델 생성.



조기 종료

- `n_iter_no_change` 하이퍼파라미터: 조기 종료 기법 지원
- 예제: 원래 500번 연속 결정트리를 훈련시켜야 하지만 검증셋에 대해 연속적으로 10번 제대로 개선되지 못하는 경우 훈련 자동 종료

```
GradientBoostingRegressor(max_depth=2,
                          learning_rate=0.05,
                          n_estimators=500,
                          n_iter_no_change=10, random_state=42)
```

- `n_iter_no_change=None`이 기본값이지만 임의의 정수로 지정되면 10% 정도의 검증셋을 매 결정트리 훈련마다 사용.
- `tol=0.0001` 허용오차 이하로 성능이 변하지 않은 경우 좋아지지 않는다고 판단

확률적 그레이디언트 부스팅

- `subsample` 하이퍼파리미터
    - 각 결정트리가 훈련에 사용할 훈련 샘플의 비율을 지정하여 학습
    - 기본값은 1
    - 예제: `subsample=0.25` 등 비율을 지정하면 지정한 비율만큼만 훈련에 사용.
- 훈련 속도 빨라짐.
- 편향 높아지지만, 분산 낮아짐.

히스토그램 그레이디언트 부스팅

- 대용량 데이터셋을 이용하여 훈련해야 하는 경우 사용
- 훈련 샘플의 특성값을 `max_bins` 개의 구간으로 분류
    - `max_bins=255`가 기본값이며 255보다 큰 값의 정수를 지정할 수 없음.
- 결정트리 CART 알고리즘의 시간 복잡도: $O(b \times m)$. 단, `b`는 실제로 사용된 구간의 수. 일반 결정트리 CART 알고리즘의 시간 복잡도는 $O(n \times m \times \log(m))$.
- 모델의 정확도는 떨어지며, 경우에 따라 과대적합을 방지하는 규제 역할 수행. 하지만 과소적합 발생 가능.

사이킷런의 히스토그램 그레이디언트 부스팅 모델

- `HistGradientBoostingRegressor` : 회귀 모델
- `HistGradientBoostingClassifier` : 분류 모델
- `GradientBoostingRegressor` , `GradientBoostingClassifier` 등과 유사하게 작동

XGBoost

- Extreme Gradient Boosting의 줄임말.
- 그레이디언트 부스팅과의 차이점
    - 결정트리 학습에 사용되는 노드 분할을 통해 낮춰야 하는 비용함수가 다름.
    - 불순도 대신 mse, logloss 등 모델 훈련의 목적에 맞는 손실 함수 사용
    - 이와 더불어 생성되는 결정트리의 복잡도도 비용함수에 추가됨. 따라서 최종적으로 생성되는 모델에 사용되는 결정트리의 복잡도를 가능한한 낮추도록 유도.
- 빠른 속도, 확장성, 이식성 뛰어남.
- 결측치 포함 데이터 처리 가능
- GPU 계산 지원

XGBoost 사용법

- 사이킷런 라이브러리에 포함되지 않음.
- `pip` 또는 `conda`를 이용하여 쉽게 설치 가능. 구글 코랩에선 이미 설치됨.

```
pip install xgboost
```

- `XGBRegressor`와 `XGBClassifier` 모델 지원. 사용법은 그레이디언트 부스팅과 유사.

```
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)
```