Stanford University

CS 329P : Practical Machine Learning (2021 Fall)

5.3 Boosting

Qingqing Huang, Mu Li, Alex Smola

https://c.d2l.ai/stanford-cs329p

Boosting

- Learn n weak learners **sequentially**, combine to reduce model bias
- At step t, repeat:
  - Evaluate the existing learners' errors ϵ_t
  - Train a weak learner $\hat{f}_t$, focus on wrongly predicted examples
    - AdaBoost: Re-sample data according to ϵ_t
    - Gradient boosting: Train learner to predict ϵ_t
  - **Additively** combining existing weak learners with $\hat{f}_t$

Gradient Boosting

- Supports arbitrary differentiable loss
- $H_t(x)$: output of combined model at timestep t, with $H_1(x) = 0$
- For each step t, repeat:
  - Train a new learner $\hat{f}_t$ on residuals: $\{(x_i, y_i - H_t(x_i))\}_{i=1,\ldots,m}$
  - Combine: $H_{t+1}(x) = H_t(x) + \eta \hat{f}_t(x)$ shrinkage parameter η for regularization
- MSE $L = \frac{1}{2}(H(x) - y)^2$, residual equals negative gradient $y - H(x) = -\frac{\partial L}{\partial H}$
  - For other loss L, learner $\hat{f}_t = \arg\min \frac{1}{2}\left(\hat{f}_t(x) + \frac{\partial L(x)}{\partial H_t}\right)^2$
- Avoid overfitting: subsampling, shrinkage, early-stopping

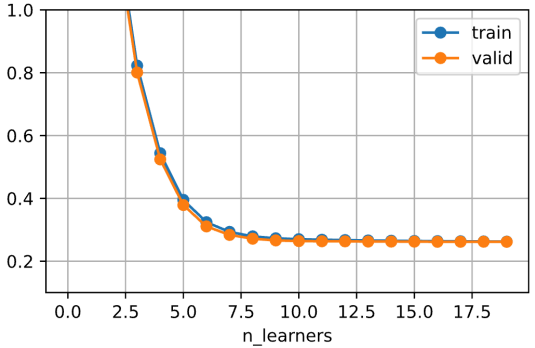
Gradient Boosting Code

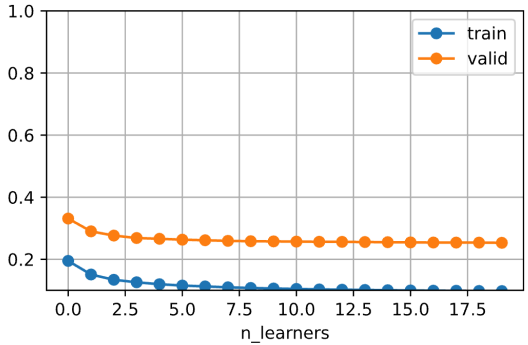
```
class GradientBoosting:
    def __init__(self, base_learner, n_learners, learning_rate):
        self.learners = [clone(base_learner) for _ in range(n_learners)]
        self.lr = learning_rate

    def fit(self, X, y):
        residual = y.copy()
        for learner in self.learners:
            learner.fit(X, residual)
            residual -= self.lr * learner.predict(X)

    def predict(self,X):
        preds = [learner.predict(X) for learner in self.learners]
        return np.array(preds).sum(axis=0) * self.lr
```

Gradient Boosting Decision Trees (GBDT)

- Use decision tree as the weak learner
  - Regularize by a small `max_depth` and randomly sampling features
- Sequentially constructing trees runs slow
  - Popular libraries use accelerated algorithms, e.g. XGBoost, lightGBM

GBDT





Random Forest