Gradient boosting

Gradient Boosting

- **Model** $$\hat{F}(x) = \sum_{i=1}^{M} \gamma_i h_i(x) + \text{const.}$$
- **Pick loss function** L(**y**,**F**(**x**))
  - L_2 or logistic or …
- **Pick base learners** h_i(**x**)
  - e.g. decision tree of specified depth
- **Optionally subsample features**
  - "stochastic gradient boosting"
- **Do stagewise estimation on F(x)**

1. Initialize model with a constant value:

$$F_0(x) = \underset{\gamma}{\arg\min} \sum_{i=1}^{n} L(y_i, \gamma).$$

2. For m = 1 to M:

    1. Compute so-called *pseudo-residuals*:

    $$r_{im} = -\left[\frac{\partial L(y_i, F(x_i))}{\partial F(x_i)}\right]_{F(x)=F_{m-1}(x)} \quad \text{for } i = 1, \ldots, n.$$

    2. Fit a base learner (e.g. tree) $h_m(x)$ to pseudo-residuals, i.e. train it using the training set $\{(x_i, r_{im})\}_{i=1}^{n}$.
    3. Compute multiplier γ_m by solving the following one-dimensional optimization problem:

    $$\gamma_m = \underset{\gamma}{\arg\min} \sum_{i=1}^{n} L\left(y_i, F_{m-1}(x_i) + \gamma h_m(x_i)\right).$$

    4. Update the model:

    $$F_m(x) = F_{m-1}(x) + \gamma_m h_m(x).$$

3. Output $F_M(x)$.

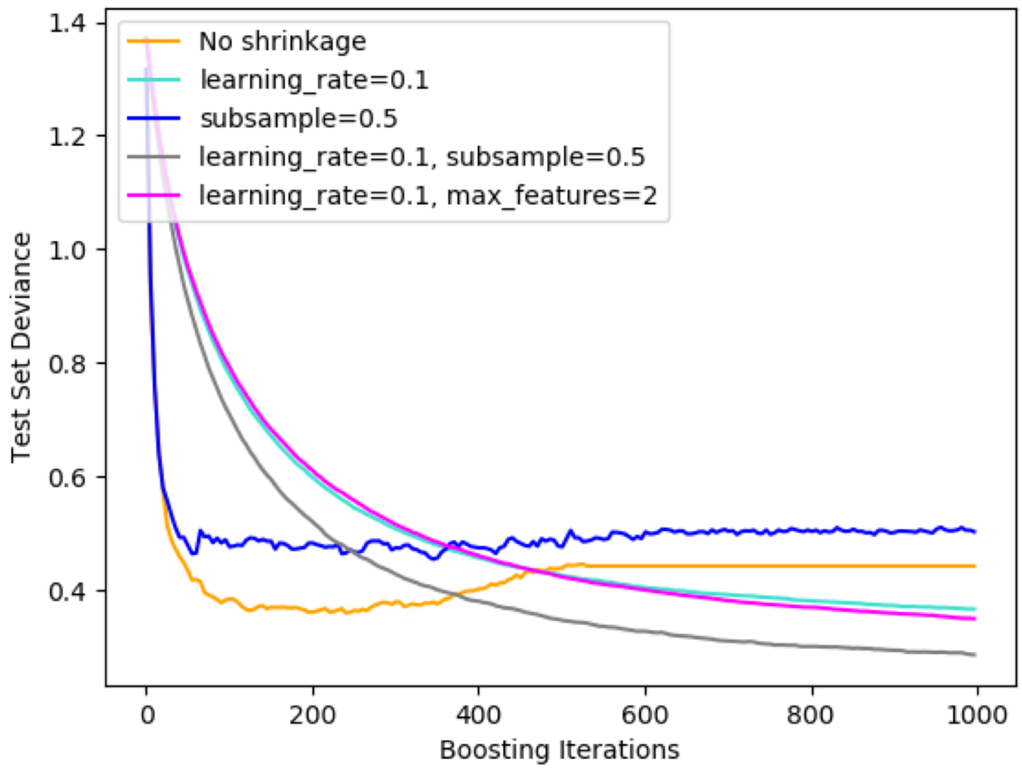
---

https://en.wikipedia.org/wiki/Gradient_boosting

Gradient Tree Boosting for Regression

- **Loss function: L_2**
- **Base learners** $h_i(\mathbf{x})$
  - Fixed depth regression tree fit on pseudo-residual
  - Gives a constant prediction for each leaf of the tree
- **Stagewise: find weights on each** $h_i(\mathbf{x})$
  - Fancy version: fit different weights for each leaf of tree

Regularization



http://scikit-learn.org/stable/auto_examples/ensemble/plot_gradient_boosting_regularization.html