Yandex

Intro into Machine Learning

Adaboost. Gradient Boosting and friends.

Clermont University,
Nov, 2017

Alexey Artemov[1,3], Andrey Ustyuzhanin[2,3]

[1] Yandex LLC [2] Yandex School of Data Analysis [3] National Research University Higher School of Econonmics

Lecture overview

- Naive boosting for regression
- Adaptive boosting
- Gradient boosting machine
- Stochastic gradient boosting
- XGBoost

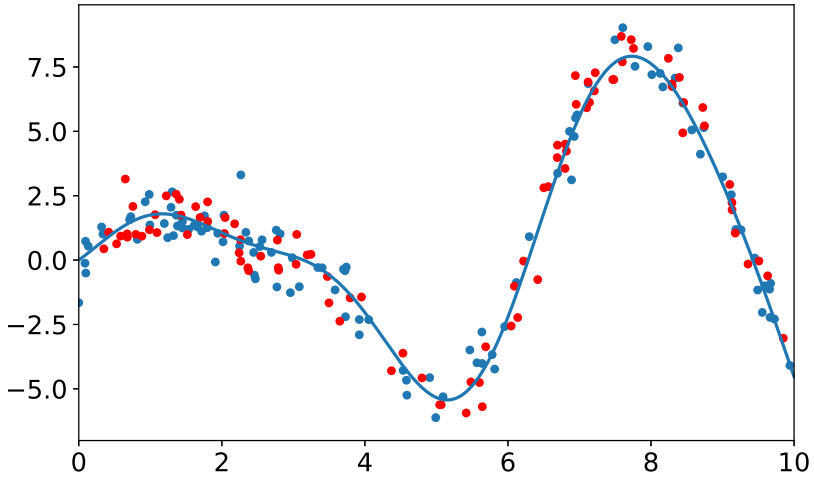
Naive boosting for regression

Boosting for regression

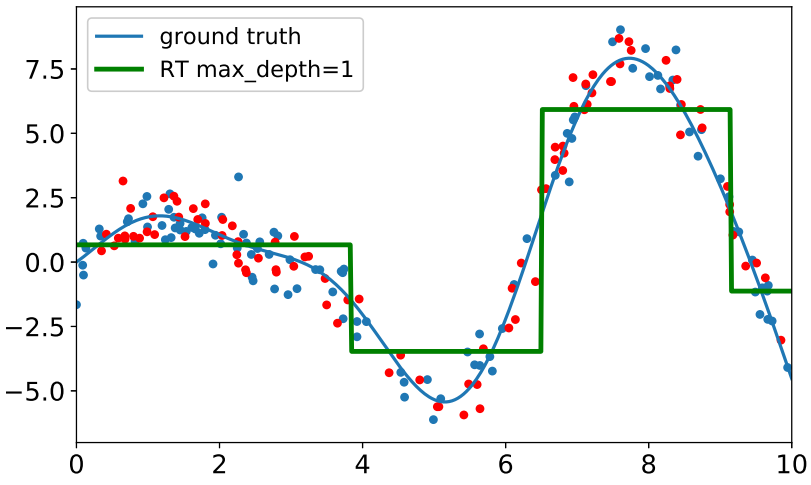
- Consider a regression problem $\frac{1}{2}\sum_{i=1}^{\ell}(h(x_i) - y_i)^2 \to \min_h$
- Search for solution in the form of weak learner composition $a_N(x) = \sum_{n=1}^{N} h_n(x)$ with weak learners $h_n \in \mathbb{H}$
- The boosting approach: add weak learners greedily
  1. Start with a "trivial" weak learner $h_0(x) = \frac{1}{\ell}\sum_{i=1}^{\ell} y_i$
  2. At step N, compute the residuals
  $$s_i^{(N)} = y_i - \sum_{n=1}^{N-1} h_n(x_i) = y_i - a_{N-1}(x_i), \qquad i = 1, \ldots, \ell$$
  3. Learn the next weak algorithm using
  $$a_N(x) := \arg\min_{h \in \mathbb{H}} \frac{1}{2}\sum_{i=1}^{\ell}(h(x_i) - s_i^{(N)})^2$$

(this implementation may be found in, e.g., `scikit-learn`)

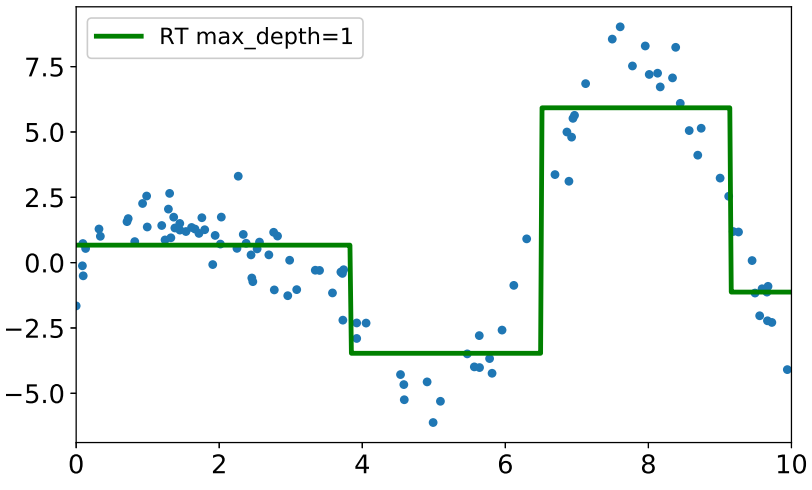
Boosting: an example regression problem



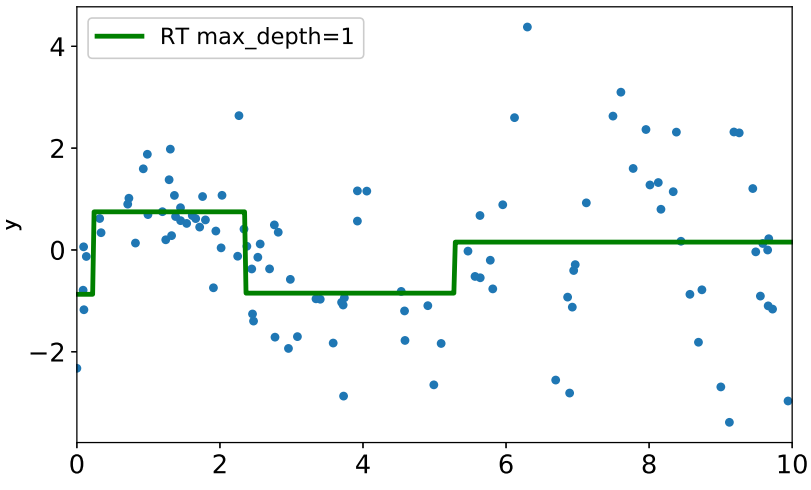
Boosting: an example regression problem



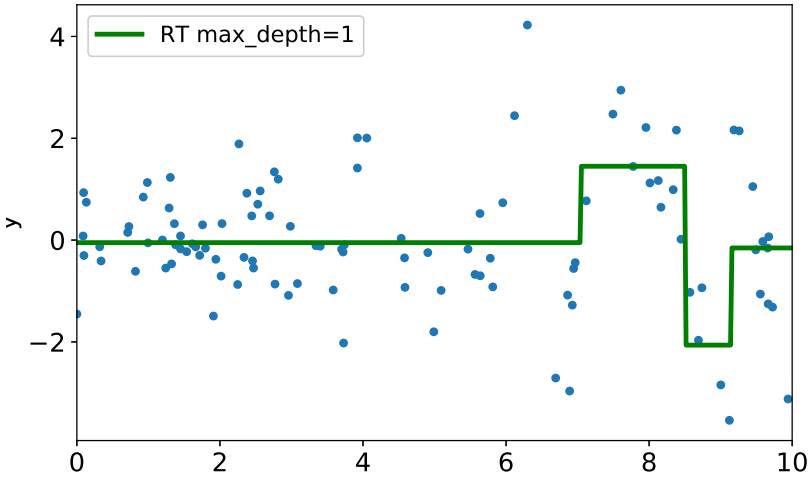
Boosting: an example regression problem



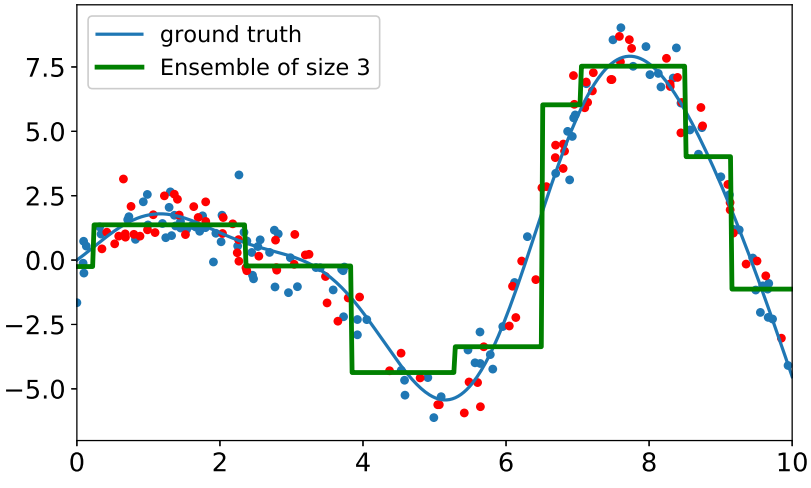
Boosting: an example regression problem



Boosting: an example regression problem



Boosting: an example regression problem

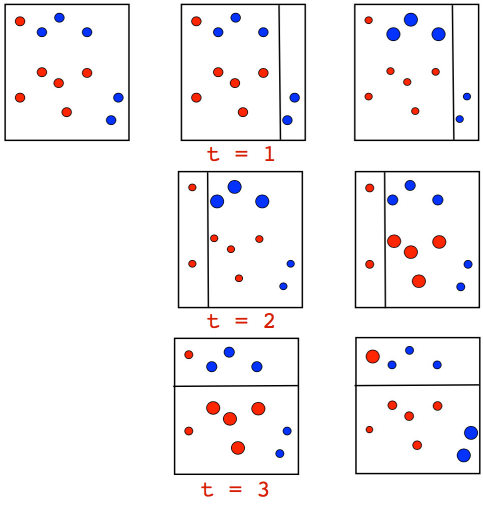


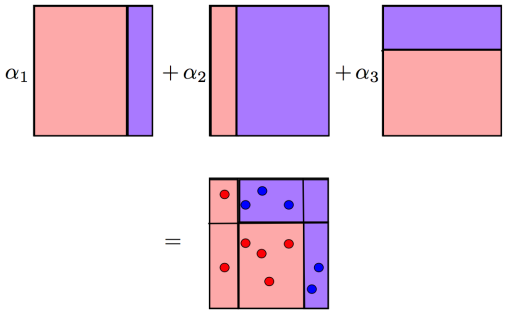
Reweighting and AdaBoost

Adaptive boosting for classification

- Training set $X^\ell = \{(\mathbf{x}_i, y_i)\}_{i=1}^\ell$, $y_i \in \{-1, +1\}$
- Search for solution in the form of a weighted sum $a_N(\mathbf{x}) = \sum_{n=1}^N \gamma_n h_n(\mathbf{x})$ with weak learners $h_n \in \mathbb{H}$
- At step N, extend a with h_N: $a_N(\mathbf{x}_i) = a_{N-1}(\mathbf{x}_i) + \gamma_N h_N(\mathbf{x}_i)$.
  How do we choose h_N and its weight γ_N?
- Measure fit quality using exponential loss $Q(a_N, X^\ell) = \sum_{i=1}^\ell \exp\{-y_i a_N(\mathbf{x}_i)\}$
- Derivation yields $h_N = \arg\min_h \sum_{i=1}^\ell (h(\mathbf{x}_i) - w_i y_i)^2$
  with weights $w_i = \exp\{-y_i a_{N-1}(\mathbf{x}_i)\}$
- Weak learner weight: minimize $\gamma_N = \arg\min_\gamma Q(a_{N-1}(\mathbf{x}_i) + \gamma h_N(\mathbf{x}_i), X^\ell)$

Adaptive boosting for classification





Picture credit: https://habrahabr.ru/company/ods/blog/327250/

Adaptive boosting for classification

[Video: AdaBoost in Action]

https://www.youtube.com/watch?v=k4G2VCuOMMg

Gradient boosting

Gradient boosting: motivation

- With $a_{N-1}(\mathbf{x})$ already built, how to find the next γ_N and h_N if

$$\sum_{i=1}^{\ell} L(y_i, a_{N-1}(\mathbf{x}_i) + \gamma h(\mathbf{x}_i)) \to \min_{\gamma, h}$$

- Recall: functions decrease in the direction of negative gradient
- View $L(y, z)$ as a function of z ($= a_N(\mathbf{x}_i)$), execute gradient descent on z
- Search for such $s_1, \ldots, s_\ell$ that

$$\sum_{i=1}^{\ell} L(y_i, a_{N-1}(\mathbf{x}_i) + s_i) \to \min_{s_1, \ldots, s_\ell}$$

- Choose $s_i = -\left.\frac{\partial L(y_i, z)}{\partial z}\right|_{z=a_{N-1}(\mathbf{x}_i)}$, approximate s_i's by $h_N(\mathbf{x}_i)$

The Gradient Boosting Machine [Friedman, 2001]

- Input:
  - Training set $X^\ell = \{(\mathbf{x}_i, y_i)\}_{i=1}^\ell$
  - Number of boosting iterations N
  - Loss function $Q(y, z)$ with its gradient $\frac{\partial Q}{\partial z}$
  - A family $\mathbb{H} = \{h(\mathbf{x})\}$ of weak learners and their associated learning procedures
  - Additional hyperparameters of weak learners (tree depth, etc.)
- Initialize GBM $h_0(\mathbf{x})$ using some simple rule (zero, most popular class, etc.)
- Execute boosting iterations $t = 1, \ldots, N$ (see next slide)
- Compose the final GBM learner: $a_N(\mathbf{x}) = \sum_{t=0}^N \gamma_i h_i(\mathbf{x})$

The Gradient Boosting Machine [Friedman, 2001]

At every iteration:

1. Compute pseudo-residuals: $s_i = -\left.\frac{\partial Q(y_i, z)}{\partial z}\right|_{z=a_{N-1}(\mathbf{x}_i)}, i = 1, \ldots, \ell$
2. Learn $h_N(\mathbf{x}_i)$ by regressing onto $s_1, \ldots, s_\ell$:

$$h_N(x) = \arg\min_{h \in \mathbb{H}} \sum_{i=1}^{\ell} \left(h(\mathbf{x}_i) - s_i\right)^2$$

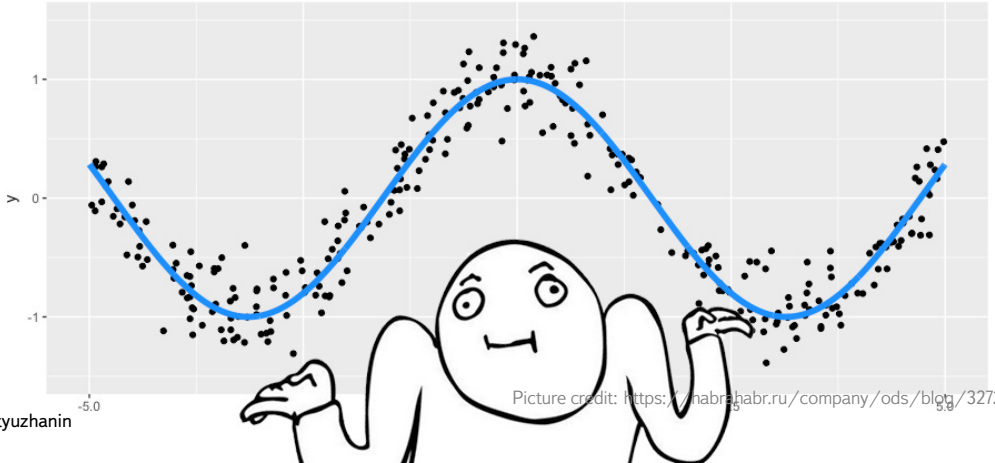
3. Find the optimal γ_N using plain gradient descent:

$$\gamma_N = \arg\min_{\gamma \in \mathbb{R}} \sum_{i=1}^{\ell} Q(y_i, a_{N-1}(\mathbf{x}_i) + \gamma h_N(\mathbf{x}_i))$$

4. Update the GBM by $a_N(\mathbf{x}_i) \leftarrow a_{N-1}(\mathbf{x}) + \gamma_N h_N(\mathbf{x})$

GBM: an example regression problem

› Consider a training set for a $X^{300} = \{(\mathbf{x}_i, y_i)\}_{i=1}^{300}$ where $x_i \in [-5, 5]$, $y_i = \cos(x_i) + \varepsilon_i$, $\varepsilon_i \sim \mathcal{N}(0, 1/5)$

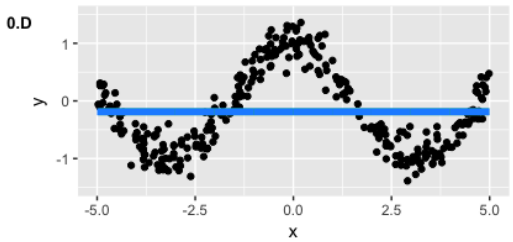


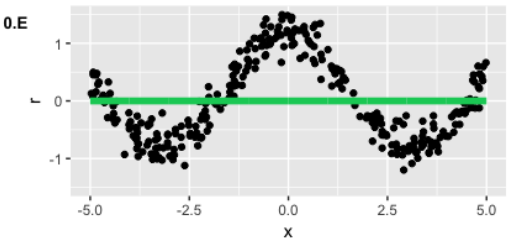
Picture credit: https://habrahabr.ru/company/ods/blog/327250

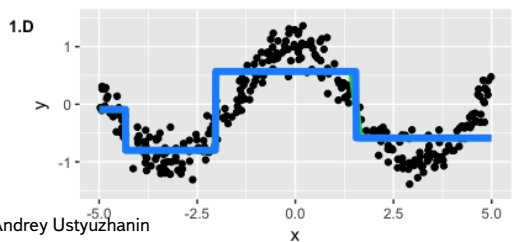
GBM: an example regression problem

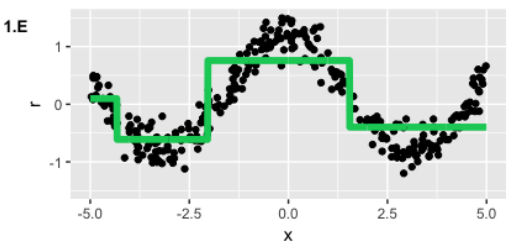
- Pick $N = 3$ boosting iterations
- Quadratic loss $Q(y, z) = (y - z)^2$
- Gradient of the quadratic loss $\frac{\partial Q(y_i, z)}{\partial z} = (y - z)$ is just redisuals
- Pick decision trees as weak learners $h_i(\mathbf{x})$
- Set 2 as the maximum depth for decision trees

GBM: an example regression problem

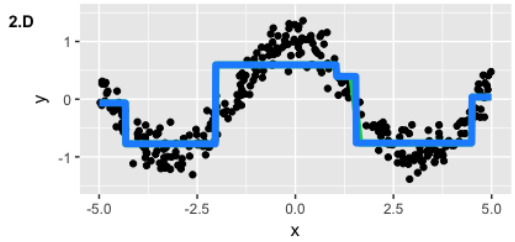


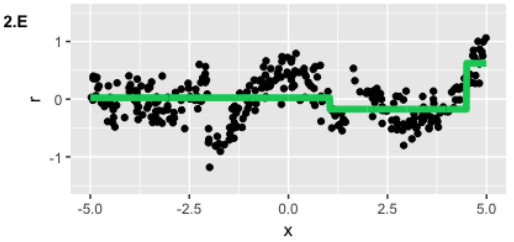


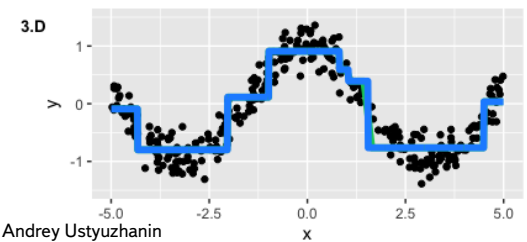


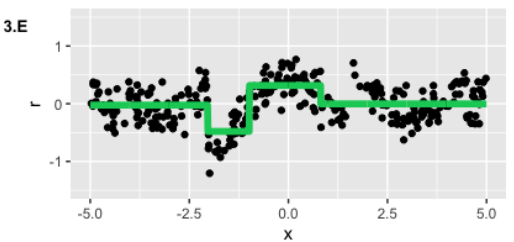


GBM: an example regression problem



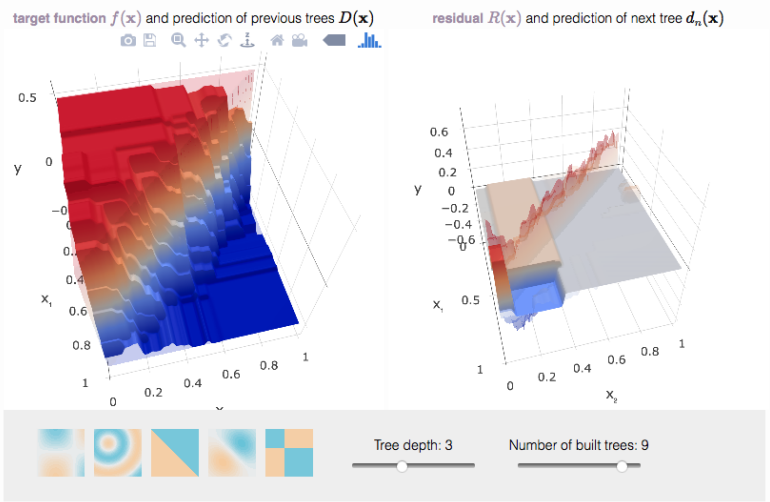






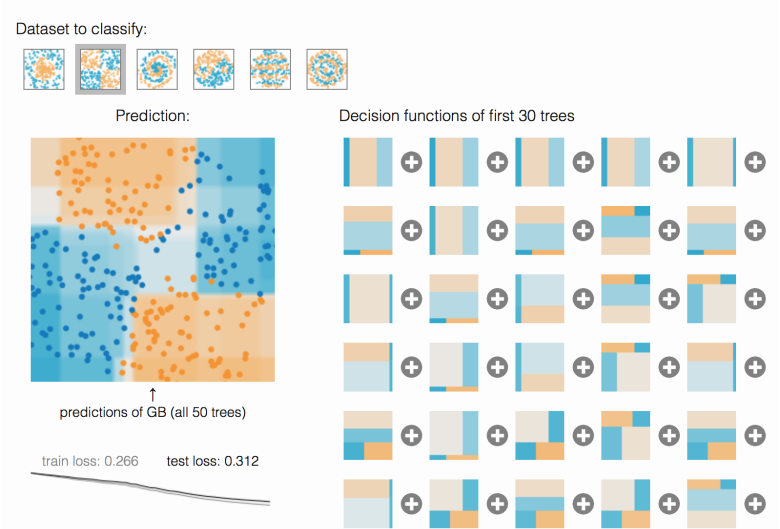
GBM: an interactive demo

http://arogozhnikov.github.io/2016/06/24/gradient_boosting_explained.html



GBM: an interactive demo

http://arogozhnikov.github.io/2016/07/05/gradient_boosting_playground.html



GBM: regulization via shrinkage

- For too simple weak learners, the negative gradient is approximated badly $\Longrightarrow$ random walk in space of samples
- For too complex weak learners, a few boosting steps may be enough for overfitting
- **Shrinkage:** make shorter steps using a learning rate $\eta \in (0, 1]$

$$a_N(\mathbf{x}_i) \leftarrow a_{N-1}(\mathbf{x}) + \eta\gamma_N h_N(\mathbf{x})$$

(effectively distrust gradient direction estimated via weak learners)

GBM: shrinkage animated

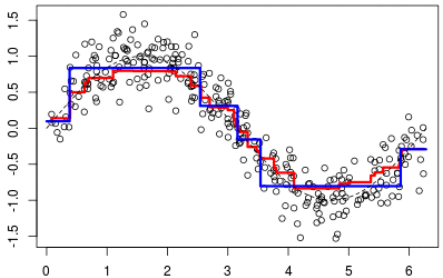


Figure: High shrinkage

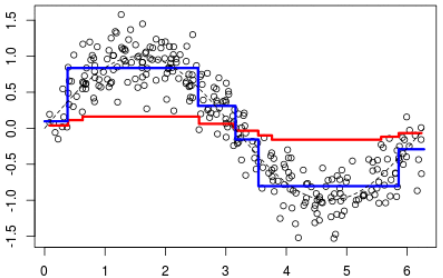
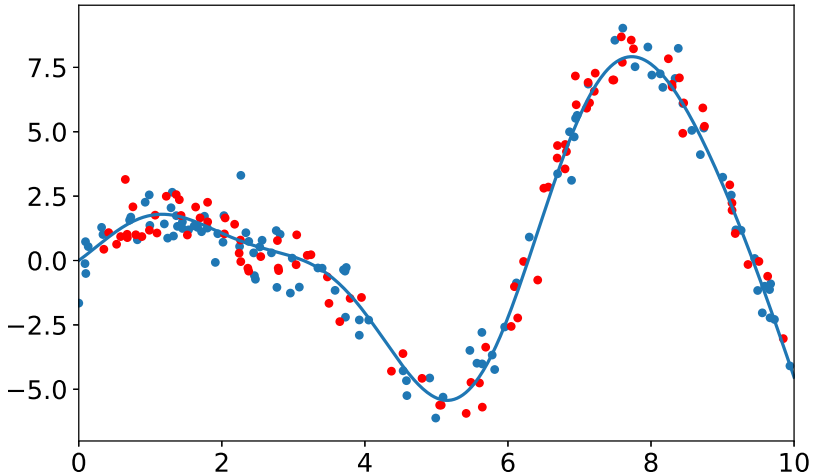
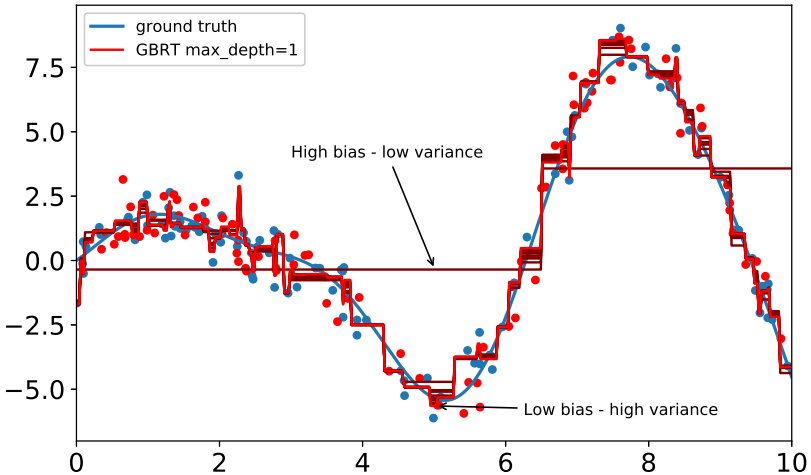


Figure: Low shrinkage

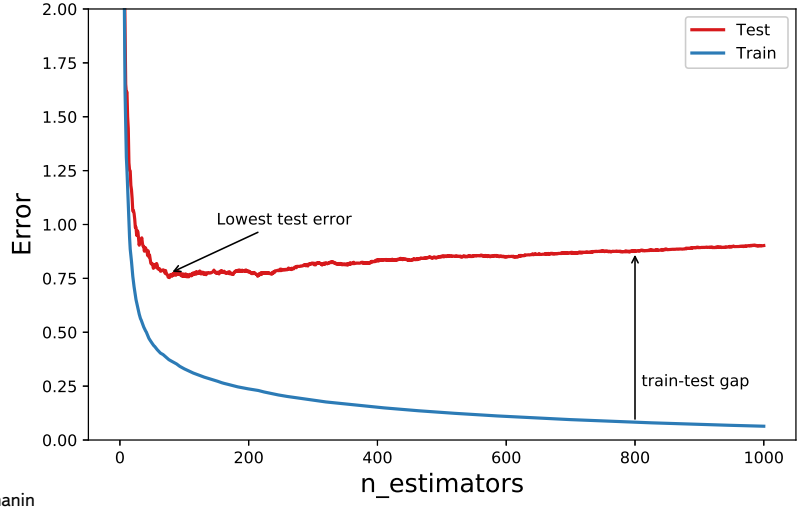
GBM: regularization approaches



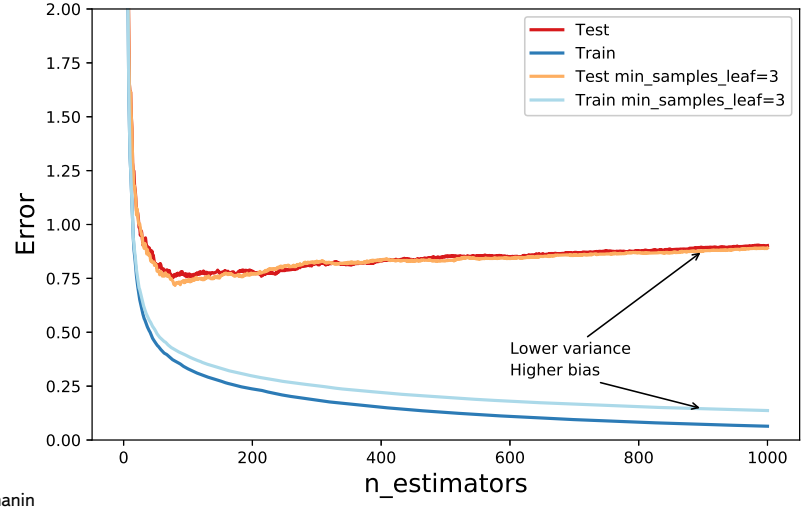
GBM: regularization approaches



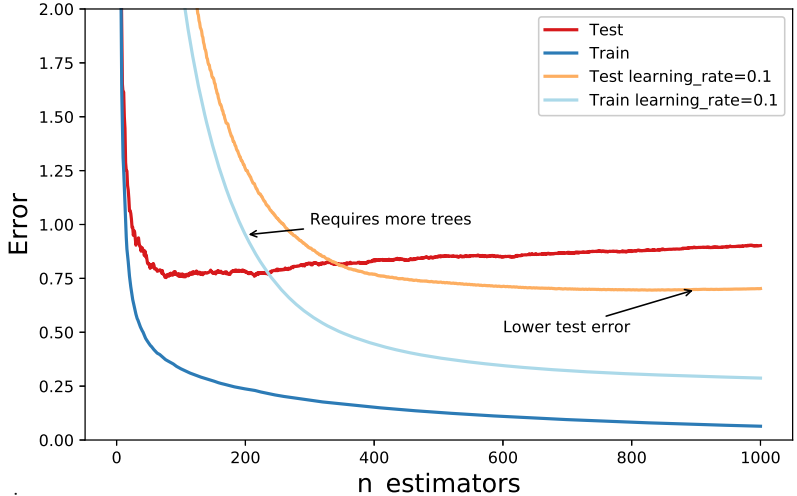
GBM: regularization approaches



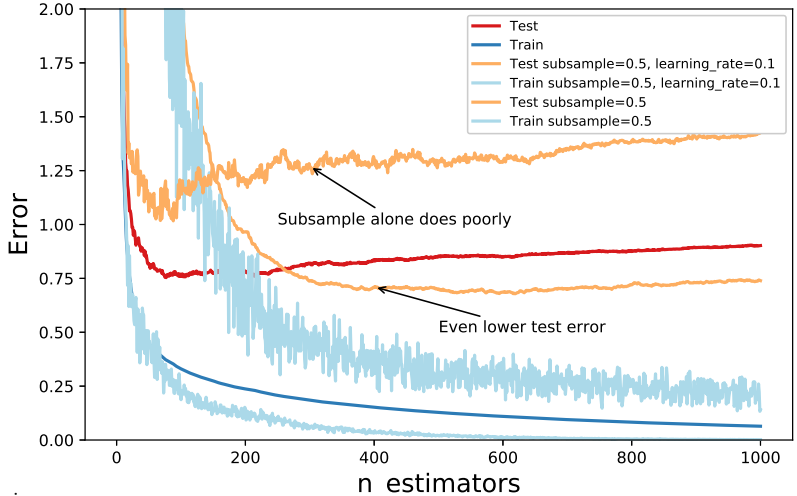
GBM: regularization approaches



GBM: regularization approaches



GBM: regularization approaches



XGBoost algorithm

Extreme Gradient Boosting

1. Approximate the descent direction constructed using second order derivatives

$$\sum_{i=1}^{\ell}\left(-s_i h(\mathbf{x}_i) + \frac{1}{2} t_i h^2(\mathbf{x}_i)\right) \to \min_h, \qquad t_i = \frac{\partial^2}{\partial z^2} L(y_i, z)\Big|_{a_{N-1}(\mathbf{x}_i)}$$

2. Penalize large leaf counts J and large leaf coefficient norm $\|b\|_2^2 = \sum_{j=1}^{J} b_j^2$

$$\sum_{i=1}^{\ell}\left(-s_i h(x_i) + \frac{1}{2} t_i h^2(x_i)\right) + \gamma J + \frac{\lambda}{2}\sum_{j=1}^{J} b_j^2 \to \min_h$$

where $b(\mathbf{x}) = \sum_{j=1}^{J} b_j [\mathbf{x} \in R_j]$

Extreme Gradient Boosting

3. Choose split $[\mathbf{x}_j < t]$ at node R to maximize

$$Q = H(R) - H(R_\ell) - H(R_r) \to \max,$$

where the impurity criterion

$$H(R) = -\frac{1}{2}\left(\sum_{(t_i,s_i)\in R} s_j\right)^2 \Bigg/ \left(\sum_{(t_i,s_i)\in R} t_j + \lambda\right) + \gamma$$

4. The stopping rule: declare the node a leaf if even the best split gives negative Q

https://xgboost.readthedocs.io/en/latest/model.html

Conclusion

- Boosting: a general meta-algorithm aimed at composing a strong hypothesis from multiple weak hypotheses
- Boosting can be applied for arbitrary losses, arbitrary problems (regression, classification) and over arbitrary weak learners
- The Gradient Boosting Machine: a general approach to boosting adding weak learners that approximate gradient of the loss function
- AdaBoost: gradient boosting with an exponential loss function resulting in reweighting training instances when adding weak learners
- XGBoost: gradient boosting with second order optimization, penalized loss and particular choice of impurity criterion