Introduction to Gradient Boosting

Jake Underland

July 7, 2022

Setup

Start with a simple setup.

- $\{x_i, y_i\}$ represents a dataset.
- N: number of training samples
- M: number of features.

Setup

In regression, we want to construct f such that

$$y_i \approx f(x_i) \text{ for all } i$$

which is to say,

$$\arg\min_f \underbrace{\frac{1}{N}\sum_i^N (y_i - f(x))^2}_{\textit{Mean Squared Error (MSE)}}$$

Setup

f can be represented as a sum of *weak learners*.

$$\underbrace{f(x)}_{\textit{final model}} = \underbrace{f_0(x) + f_1(x) + f_2(x) + \cdots + f_{\max}(x)}_{\textit{weak learners}}$$

Steps to training the model:

$$\begin{aligned} S_0(x) &= f_0(x) \\ S_1(x) &= f_0(x) + f_1(x) \\ S_2(x) &= f_0(x) + f_1(x) + f_2(x) \\ &\vdots \\ S_{k+1}(x) &= S_k(x) + f_{k+1}(x) \end{aligned}$$

S represents the model at each step, S_{max} being the end product.

Optimizing

At every step S we want to approximate y (training data).

$$S_{k+1}(x) = S_k(x) + f_{k+1}(x) \approx y$$

Reordering with respect to f_{k+1} gives us:

$$f_{k+1}(x) \approx \underbrace{y - S_k(x)}_{Residual}$$

Thus, for every step S_{k+1} we want to regress x on the residuals of the previous step, or S_k, to get f_{k+1}.

In Gradient Boosting, the model approaches f step by step.

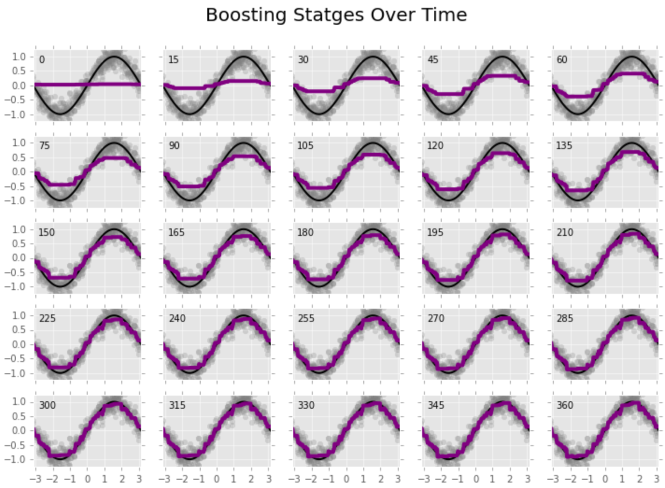
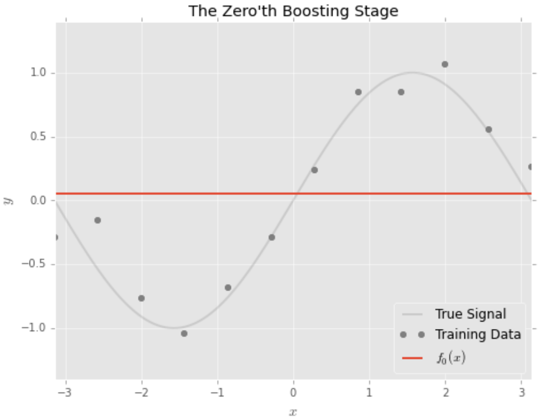


Figure: borrowed from Matthew Drury

Following the steps

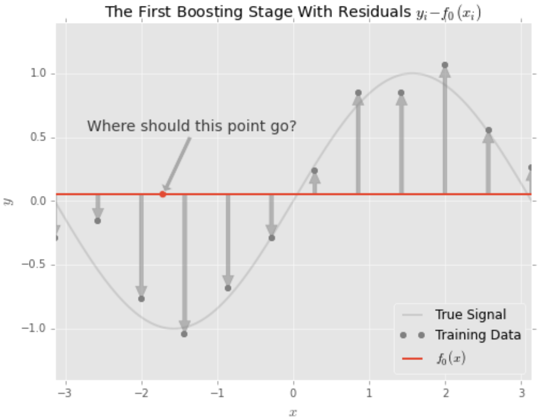
We start with $S_0(x) = f_0(x)$. Since our loss function is MSE, our choice for $f_0(x)$ should be:

$$f_0(x) = \frac{1}{N} \sum_i y_i$$



Adjust model in direction of residual $y_i - f_0(x_i)$
The residuals are only defined at the points where training data exists...

$\rightarrow$ Use *regression*!

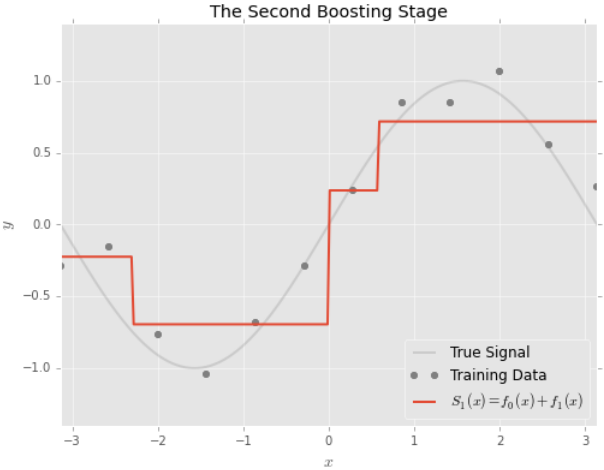


Calculating $f_1(x)$

Let residuals be the new dataset (y_i'). If we use a tree regressor, we can estimate the first tree $f_1(x)$.

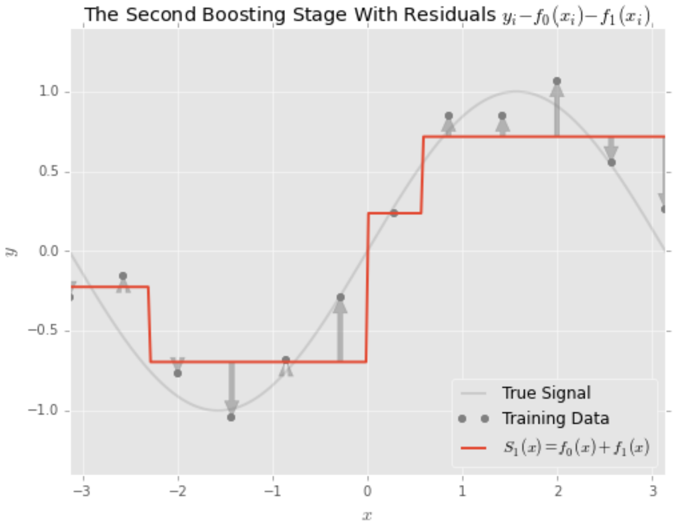
Updating the model

$$S_1(x) = f_0(x) + f_1(x) \leftarrow \text{Model fit to residuals!}$$



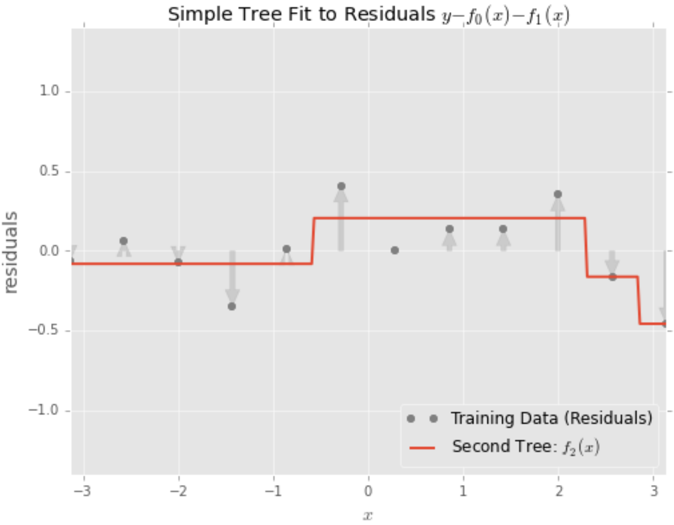
Another Step

Calculate residuals of present model...

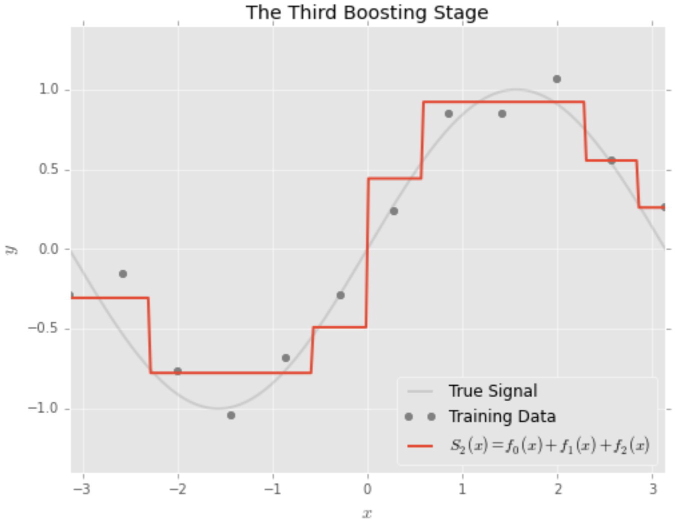


Create new training dataset $\{x_i, y'_i\}$ so that y'_i are residuals.

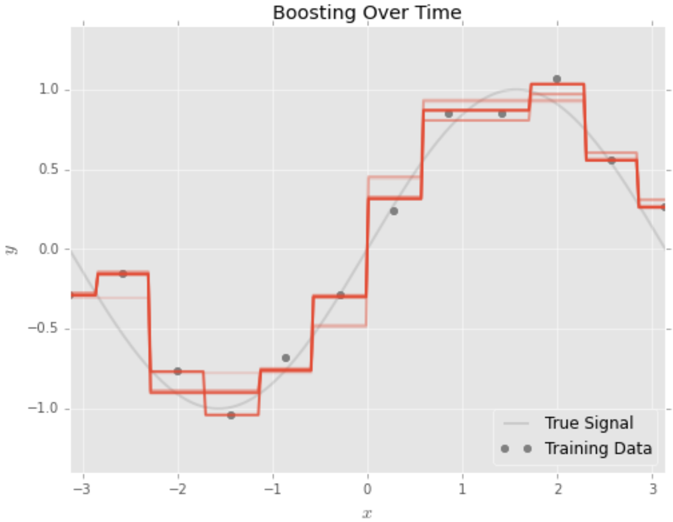
Estimate weak learner f_2 to add to model



Update model!



Gradually refine



Loss function

In the above example we used MSE as the loss function

Any loss function $L\left(y, S(x)\right)$ can be used $\downarrow$

$$S_{k+1}(x) = S_k(x) + \arg\min_f \sum_{i=1}^{N} L(y_i, S_k(x_i) + f_{k+1}(x_i))$$

BUT global minimum at each iteration is computationally expensive...

$\rightarrow$ use **gradient descent**

$$S_{k+1}(x) = S_k(x) - \left[\sum_{i=1}^{N} \nabla_{S_k} L(y_i, S_k(x_i))\right]$$

Gradient Boosting!

Gradient Descent

For any differentiable function $L(x)$

Inputs: Function L.

Outputs: x^* that minimizes L.

Algorithm: Repeat $x_{i+1} = x_i - \lambda \nabla L(x_i)$ until x converges to x^*.

Gradient Boosting

$$S_{k+1}(x) = S_k(x) - \left[\sum_{i=1}^{N} \nabla_{S_k} L(y_i, S_k(x_i))\right]$$

Gradient Boosting!

Adding weights

We can further increase accuracy by adding weights to the weak learners when updating the model.

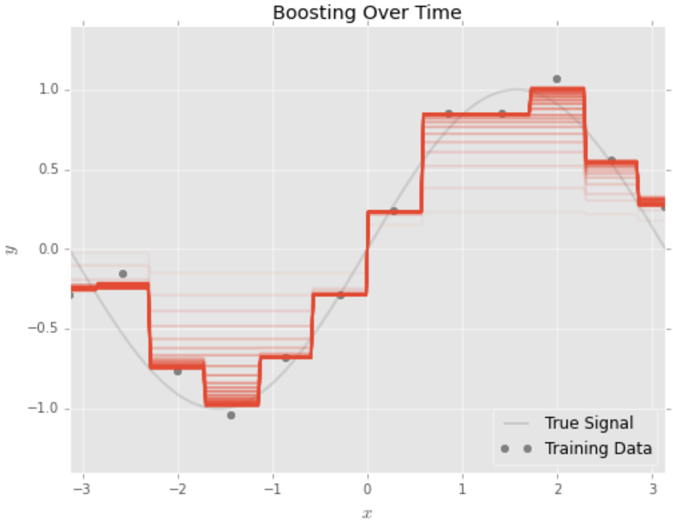
$$S_{k+1}(x) = S_k(x) + \lambda_{k+1} f_{k+1}(x)$$
$$\left(= S_k(x) - \lambda_{k+1} \sum_{i=1}^{N} \nabla_{S_k} L(y_i, S_k(x_i)) \right)$$

Where

$$\begin{aligned} \lambda_{k+1} &= \arg\min_{\lambda} \sum_{i=1}^{N} L\left(y_i, S_{k+1}(x_i)\right) \\ &= \arg\min_{\lambda} \sum_{i=1}^{N} L\left(y_i, S_k(x_i) + \lambda_{k+1} f_{k+1}(x_i)\right) \end{aligned}$$

Weights can be constant, in which case it is a *learning rate*

Smoother approximation using weights



By the way

Applying Gradient Descent to MSE yields

$$\begin{aligned} x_{k+1} &= x_k - \lambda \nabla_x L(x_k, y) \\ &= x_k - \lambda \nabla_x \frac{\partial}{\partial x}\left(\frac{1}{2}(y - x_k)^2\right) \\ &= x_k + \lambda (y - x_k) \end{aligned}$$

Meaning under MSE, GB simply moves gradually in the direction of the residuals.

Review

Inputs: A training data set $\{x_i, y_i\}$, and, optionally, a learning rate λ to replace weights.
Returns: A function f such that $f(x_i) \approx y_i$.

- Initialize $S_0(x) = f_0(x) = \frac{1}{N}\sum_i y_i$.
- Iterate (parameter k) until satisfied:
  - Create the working data set $W_k = \{x_i, y_i - S_k(x_i)\}$.
  - Fit a regression tree to W_k, minimizing least squares. Call this tree f_k.
  - Set $S_{k+1}(x) = S_k(x) + \lambda f_k(x)$.
- Return $f(x) = f_0(x) + f_1(x) + f_2(X) + \cdots + f_{\max}(x)$.