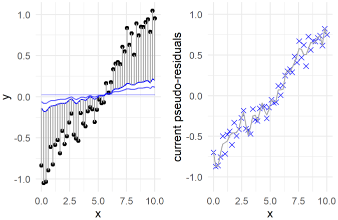
Introduction to Machine Learning

Gradient Boosting - Illustration



Learning goals

- Understand impact of different loss functions and
- Understand impact of different base learners for regression

GRADIENT BOOSTING ILLUSTRATION - GAM

We now compare different loss functions and base learners. We start with a GAM as base learner and compare the $L2$ loss with the $L1$ loss.

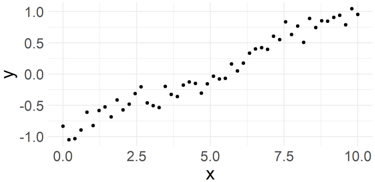
Reminder: Pseudo-residuals

- $L2$: $\tilde{r}(f) = r(f) = y - f(\mathbf{x})$
- $L1$: $\tilde{r}(f) = sign(y - f(\mathbf{x}))$

We consider a regression task with a single feature x and target y, with the following true underlying relationship:

$$y^{(i)} = -1 + 0.2 \cdot x^{(i)} + 0.1 \cdot sin(x^{(i)}) + \epsilon^{(i)}$$

$$\text{with } n = 50 \text{ and } \epsilon^{(i)} \sim \mathcal{N}(0, 0.1) \quad \forall i \in \{1, \ldots, n\}$$

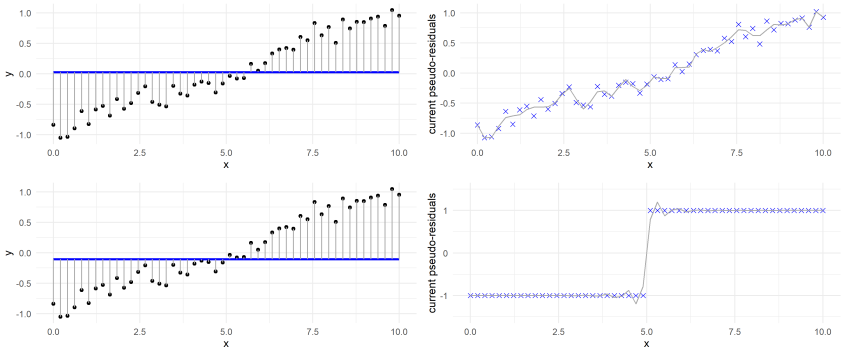


GRADIENT BOOSTING ILLUSTRATION - GAM

1. We start with the simplest model, the optimal constant – mean of the target variable in the case of $L2$ loss and median in the case of $L1$ loss.
2. We improve the model by calculating the pointwise pseudo-residuals on the training data, and fit a GAM on the residuals.
    1. The GAM base learners model the conditional mean via cubic B-splines with 40 knots.
    2. In each step, the GAM fitted on the current pseudo-residuals is multiplied by a constant learning rate of 0.2 and added to the previous model.
    3. This procedure is repeated multiple times.

GAM WITH $L2$ VS $L1$ LOSS

Top: $L2$ loss, bottom: $L1$ loss

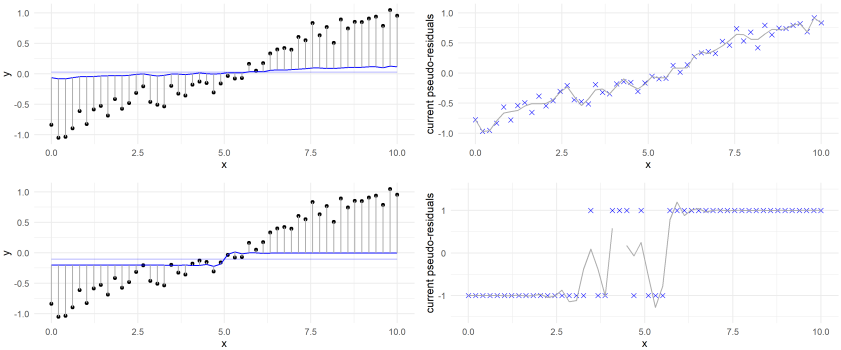


Iteration 1

The nature of the pseudo-residuals affects gradual model fit: as $L1$ only considers residuals' sign, the corresponding base learners are less affected by very large or small values compared to $L2$ and hence lead to more moderate changes.

GAM WITH $L2$ VS $L1$ LOSS

Top: $L2$ loss, bottom: $L1$ loss

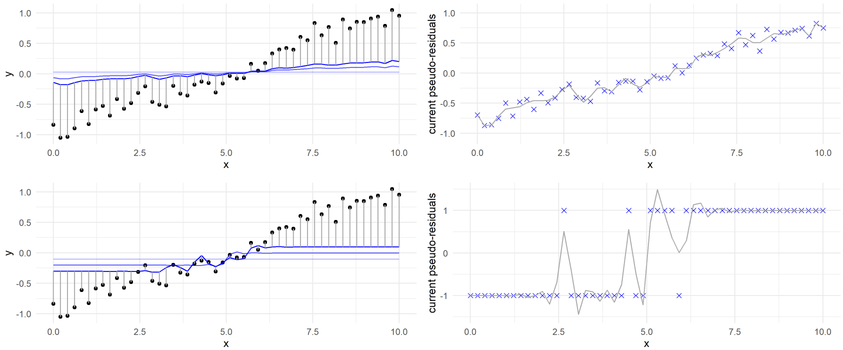


Iteration 2

The nature of the pseudo-residuals affects gradual model fit: as $L1$ only considers residuals' sign, the corresponding base learners are less affected by very large or small values compared to $L2$ and hence lead to more moderate changes.

GAM WITH $L2$ VS $L1$ LOSS

Top: $L2$ loss, bottom: $L1$ loss

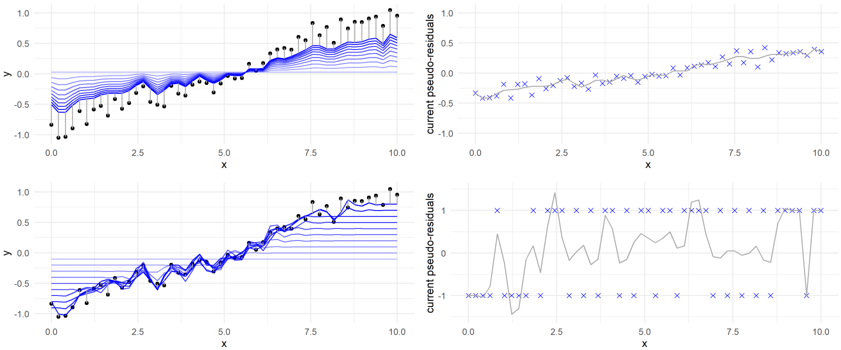


Iteration 3

The nature of the pseudo-residuals affects gradual model fit: as $L1$ only considers residuals' sign, the corresponding base learners are less affected by very large or small values compared to $L2$ and hence lead to more moderate changes.

GAM WITH $L2$ VS $L1$ LOSS

Top: $L2$ loss, bottom: $L1$ loss

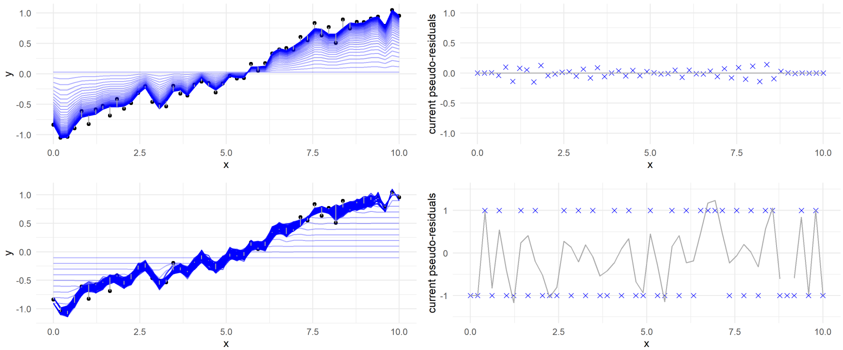


Iteration 10

The nature of the pseudo-residuals affects gradual model fit: as $L1$ only considers residuals' sign, the corresponding base learners are less affected by very large or small values compared to $L2$ and hence lead to more moderate changes.

GAM WITH $L2$ VS $L1$ LOSS

Top: $L2$ loss, bottom: $L1$ loss

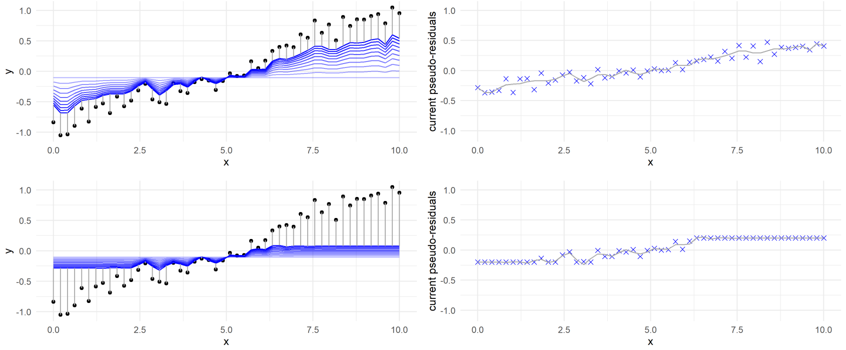


Iteration 100

The nature of the pseudo-residuals affects gradual model fit: as $L1$ only considers residuals' sign, the corresponding base learners are less affected by very large or small values compared to $L2$ and hence lead to more moderate changes.

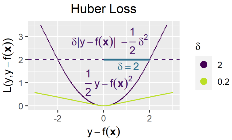
GAM WITH HUBER LOSS

We can also use Huber loss, which is closer to $L2$ for large δ values and closer to $L1$ for smaller δ values. Top: $\delta = 2$, bottom: $\delta = 0.2$.



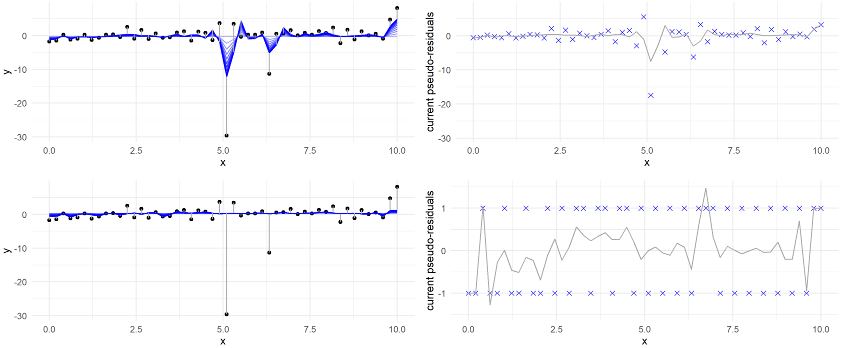
Iteration 10

We see how for smaller δ (bottom) pseudo-residuals are often effectively bounded, resulting in $L1$-like behavior, while the upper plot more closely resembles $L2$ loss.



GAM WITH OUTLIERS

Instead of normally distributed noise we can assume a t-distribution, leading to outliers in the observed target values. Top: $L2$, bottom: $L1$.

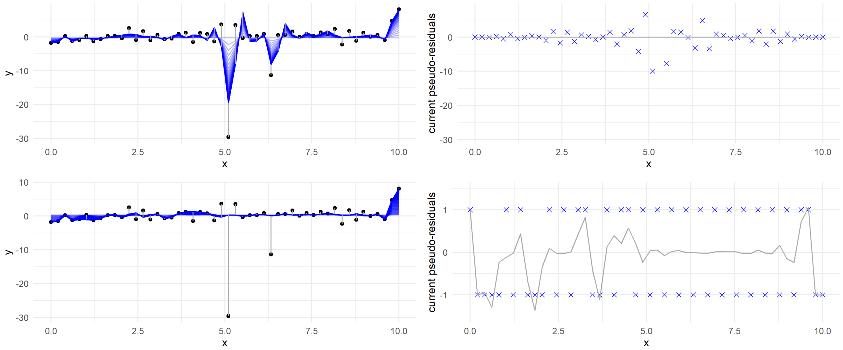


Iteration 10

*L*2 loss is affected by outliers rather strongly, whereas *L*1 solely considers residuals' sign and not their magnitude, resulting in a more robust model.

GAM WITH OUTLIERS

Instead of normally distributed noise we can assume a t-distribution, leading to outliers in the observed target values. Top: $L2$, bottom: $L1$.

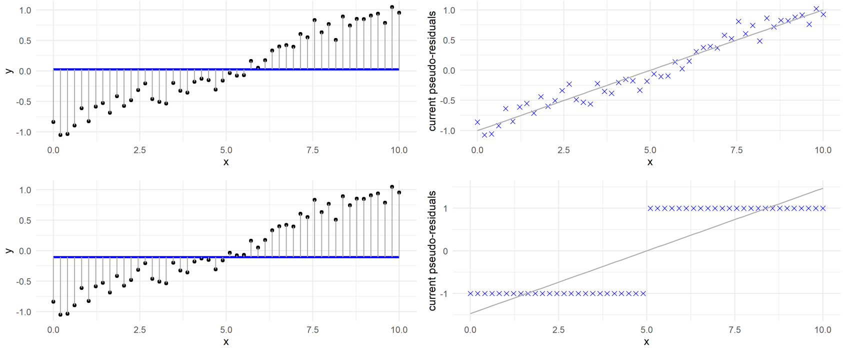


Iteration 100

$L2$ loss is affected by outliers rather strongly, whereas $L1$ solely considers residuals' sign and not their magnitude, resulting in a more robust model.

LM WITH $L2$ VS $L1$ LOSS

Instead of using a GAM as base learner we now use a simple linear model. Top: $L2$, bottom: $L1$.

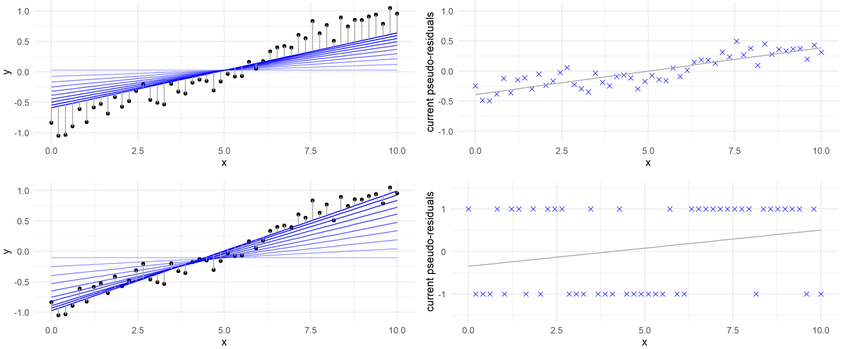


Iteration 1

For $L2$, as $\tilde{r}(f) = r(f)$, we find the optimal model in the very first iteration; only the multiplicative learning rate slows down optimization. In the $L1$ case the base learner LMs fit pseudo-residuals that differ from model residuals, leading to a less monotonic optimization path.

LM WITH $L2$ VS $L1$ LOSS

Instead of using a GAM as base learner we now use a simple linear model. Top: $L2$, bottom: $L1$.

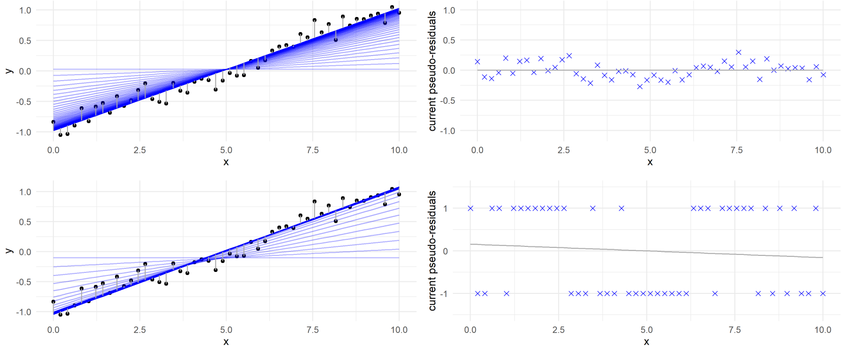


Iteration 10

For $L2$, as $\tilde{r}(f) = r(f)$, we find the optimal model in the very first iteration; only the multiplicative learning rate slows down optimization. In the $L1$ case the base learner LMs fit pseudo-residuals that differ from model residuals, leading to a less monotonic optimization path.

LM WITH $L2$ VS $L1$ LOSS

Instead of using a GAM as base learner we now use a simple linear model. Top: $L2$, bottom: $L1$.



Iteration 100

For $L2$, as $\tilde{r}(f) = r(f)$, we find the optimal model in the very first iteration; only the multiplicative learning rate slows down optimization. In the $L1$ case the base learner LMs fit pseudo-residuals that differ from model residuals, leading to a less monotonic optimization path.

LM: GB VS GD

- As we have seen, boosting with LMs and $L2$ loss simply approaches the closed-form OLS solution with a speed determined by the learning rate β.
- This is perfectly equivalent to fitting an LM via **gradient descent**, as a gradient step in parameter space is a gradient step in function space for this specific case.
- Recall the parameter update for GD with learning rate β: $\boldsymbol{\theta}^{[m+1]} \leftarrow \boldsymbol{\theta}^{[m]} - \beta \cdot \nabla_{\boldsymbol{\theta}^{[m]}} \mathcal{R}_{\text{emp}}(\boldsymbol{\theta}^{[m]}) = \boldsymbol{\theta}^{[m]} + \beta(-\mathbf{X}^T\mathbf{y} + \mathbf{X}^T\mathbf{X}\boldsymbol{\theta}^{[m]})$.
- Now compute the update for a boosted LM with current model $\mathbf{X}\boldsymbol{\theta}^{[m]}$ (note that adding a linear base learner to an LM is equivalent to summing parameters):

$$
\begin{aligned}
\frac{\partial}{\partial \boldsymbol{\theta}^{[m+1]}} \left\| (\mathbf{y} - \mathbf{X}\boldsymbol{\theta}^{[m]}) - \mathbf{X}\boldsymbol{\theta}^{[m+1]}) \right\|_2^2 &= 0 \\
-2\mathbf{X}^T(\mathbf{y} - \mathbf{X}\boldsymbol{\theta}^{[m]}) + 2\mathbf{X}^T\mathbf{X}\boldsymbol{\theta}^{[m+1]} &= 0 \\
\boldsymbol{\theta}^{[m+1]} &= (\mathbf{X}^T\mathbf{X})^{-1}\mathbf{X}^T(\mathbf{y} - \mathbf{X}\boldsymbol{\theta}^{[m]}) \\
\boldsymbol{\theta}^{[m+1]} &= (\mathbf{X}^T\mathbf{X})^{-1}\mathbf{X}^T\mathbf{y} - (\mathbf{X}^T\mathbf{X})^{-1}\mathbf{X}^T\mathbf{X}\boldsymbol{\theta}^{[m]} \\
\boldsymbol{\theta}^{[m+1]} &= (\mathbf{X}^T\mathbf{X})^{-1}\mathbf{X}^T\mathbf{y} - \boldsymbol{\theta}^{[m]} \\
\boldsymbol{\theta}^{[m+1]} &= -\mathbf{X}^T\mathbf{y} + \mathbf{X}^T\mathbf{X}\boldsymbol{\theta}^{[m]}.
\end{aligned}
$$

$$
\Rightarrow \hat{f}^{[m+1]} = \mathbf{X}\tilde{\boldsymbol{\theta}}^{[m+1]} = \mathbf{X}\left(\boldsymbol{\theta}^{[m]} + \beta(-\mathbf{X}^T\mathbf{y} + \mathbf{X}^T\mathbf{X}\boldsymbol{\theta}^{[m]})\right).
$$