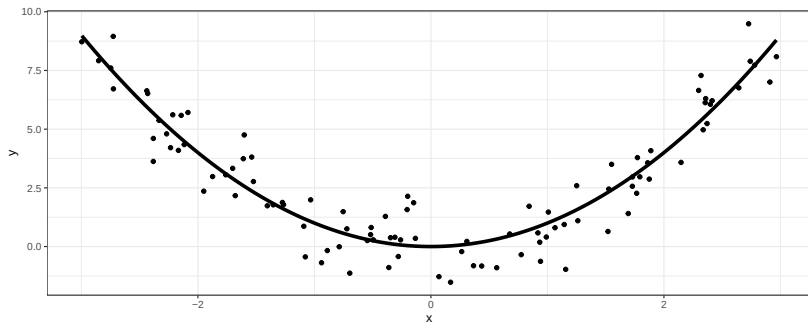


Boosting

Ryan Miller

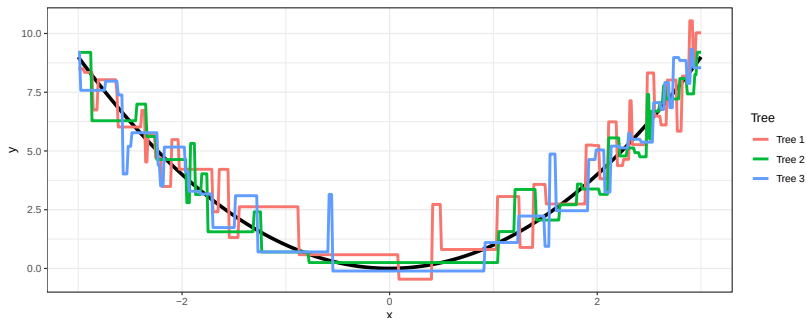
Introduction

In our random forests lecture, we considered data where the true relationship is $Y = X^2 + \text{error}$:



Introduction (continued)

The random forest algorithm aggregated the predictions of different base learners trained on bootstrapped samples:



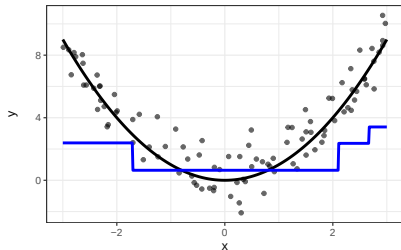
Each base learner's prediction function $\hat{f}_i()$ is high variance estimate of $f()$, but their individual weaknesses are averaged out

Boosting

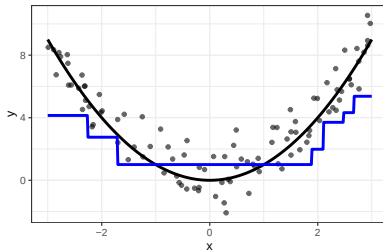
- ▶ Consider a single tree, $\hat{f}_1(\mathbf{x})$, which leaves residuals:
 $\mathbf{r}_1 = \mathbf{y} - \hat{f}_1(\mathbf{x})$
 - ▶ Boosting will fit a second tree using $\mathbf{r}_1$ as the outcome variable, yielding $\hat{f}_2()$
- ▶ The prediction function is updated such that
 $\hat{f}() = \hat{f}_1(\mathbf{x}) + \eta \hat{f}_2(\mathbf{x})$, where η is the **learning rate**
 - ▶ This process is repeated, using $\mathbf{r}_2 = \mathbf{y} - (\hat{f}_1(\mathbf{x}) + \eta \hat{f}_2(\mathbf{x}))$ as the outcome to estimate $\hat{f}_3()$
 - ▶ We end up with a final model looking like:
 $\hat{f}(\mathbf{x}) = \hat{f}_1(\mathbf{x}) + \eta \hat{f}_2(\mathbf{x}) + \eta \hat{f}_3(\mathbf{x}) + \dots + \eta \hat{f}_m(\mathbf{x})$
- ▶ A smaller learning rate limits how much each new tree contributes, with values typically ranging from 0.01 to 0.3

Boosting

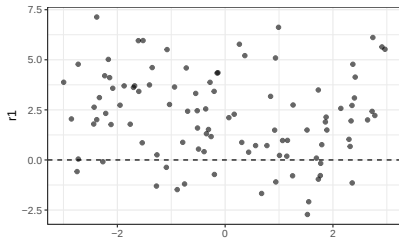
Tree 1



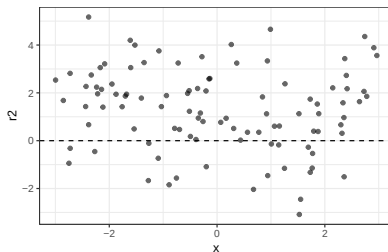
Trees 1 + 2



Residuals After Tree 1

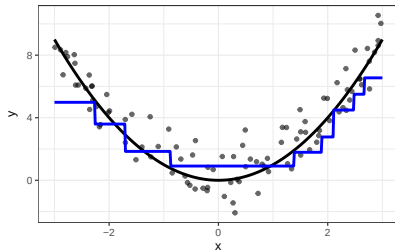


Residuals After Tree 2

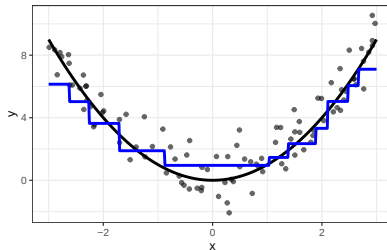


Boosting

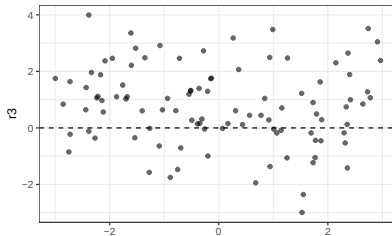
Trees 1 + 2 + 3



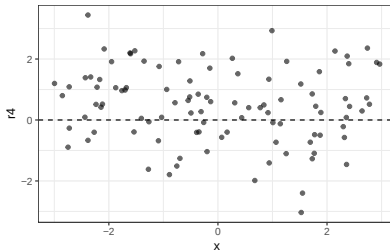
Trees 1 + 2 + 3 + 4



Residuals After Tree 3



Residuals After Tree 4



Boosting vs. Random Forest

Random Forest:

- ▶ Deep trees
- ▶ Trees built independently
- ▶ Averaging to reduce variance
- ▶ Adding more trees $\neq$ overfitting

Boosting:

- ▶ Shallow trees
- ▶ Trees built sequentially
- ▶ Trained on prior residuals to reduce bias
- ▶ Adding more trees *can* lead to overfitting

Modern Boosting Algorithms

Boosting is generally regarded as a state of the art algorithm for tabular data, with several libraries you should be aware of:

1. XGBoost - “Extreme Gradient Boost”, a highly optimized and widely used boosting library known for strong predictive performance and excellent documentation.
2. LightGBM - “Light Gradient Boosted Machines”, a boosting implementation developed by Microsoft for speed and memory efficiency.
3. CatBoost - A boosting library specifically designed to work with categorical predictors, requiring little or no preprocessing.

Modern Boosting Algorithms

Tuning a modern boosting algorithm generally focuses on the following hyperparameters:

- ▶ `n_estimators` - the number of boosting iterations
- ▶ `learning_rate` - how much each new tree contributes, or η
- ▶ `max_depth` - maximum depth of the base learners
- ▶ `subsample` - fraction of training observations used by each tree, can improve computational efficiency and reduce overfitting by introducing randomness
- ▶ `colsample_bytree` - similar to `max_features` in random forests, helps prevent overfitting by randomly sampling features

A common strategy is to use shallow trees, a small learning rate, and many boosting iterations

Why is it called “gradient boosting”? For numeric outcomes assessed using squared error, the residuals are the *negative gradient* of the loss function with respect to the predicted values, $\hat{\mathbf{y}}$:

$$\text{Loss} = \frac{1}{n} \sum_{i=1}^n (y_i - \hat{y}_i)^2 = \frac{1}{n} (\mathbf{y} - \hat{\mathbf{y}})^T (\mathbf{y} - \hat{\mathbf{y}})$$

Taking the derivative with respect to the predicted values gives:

$$\frac{\partial \text{Loss}}{\partial \hat{\mathbf{y}}} = -\frac{2}{n} (\mathbf{y} - \hat{\mathbf{y}}) = -\frac{2}{n} \mathbf{r}$$

So, for squared error, boosting is just sequentially fitting trees to the negative gradient at each iteration. . .

What to Know for the Next Quiz

1. Differences between random forests and boosting
2. Basic concepts of boosting:
 - ▶ The first tree captures the largest, most obvious patterns.
 - ▶ Each new tree is trained to explain the residual errors left by the current ensemble.
 - ▶ The final prediction is the sum of many small contributions.
3. Learning rate:
 - ▶ Controls how much each new tree contributes to the ensemble.
 - ▶ Smaller learning rates require more boosting iterations.
 - ▶ Larger learning rates learn more quickly but may overfit.