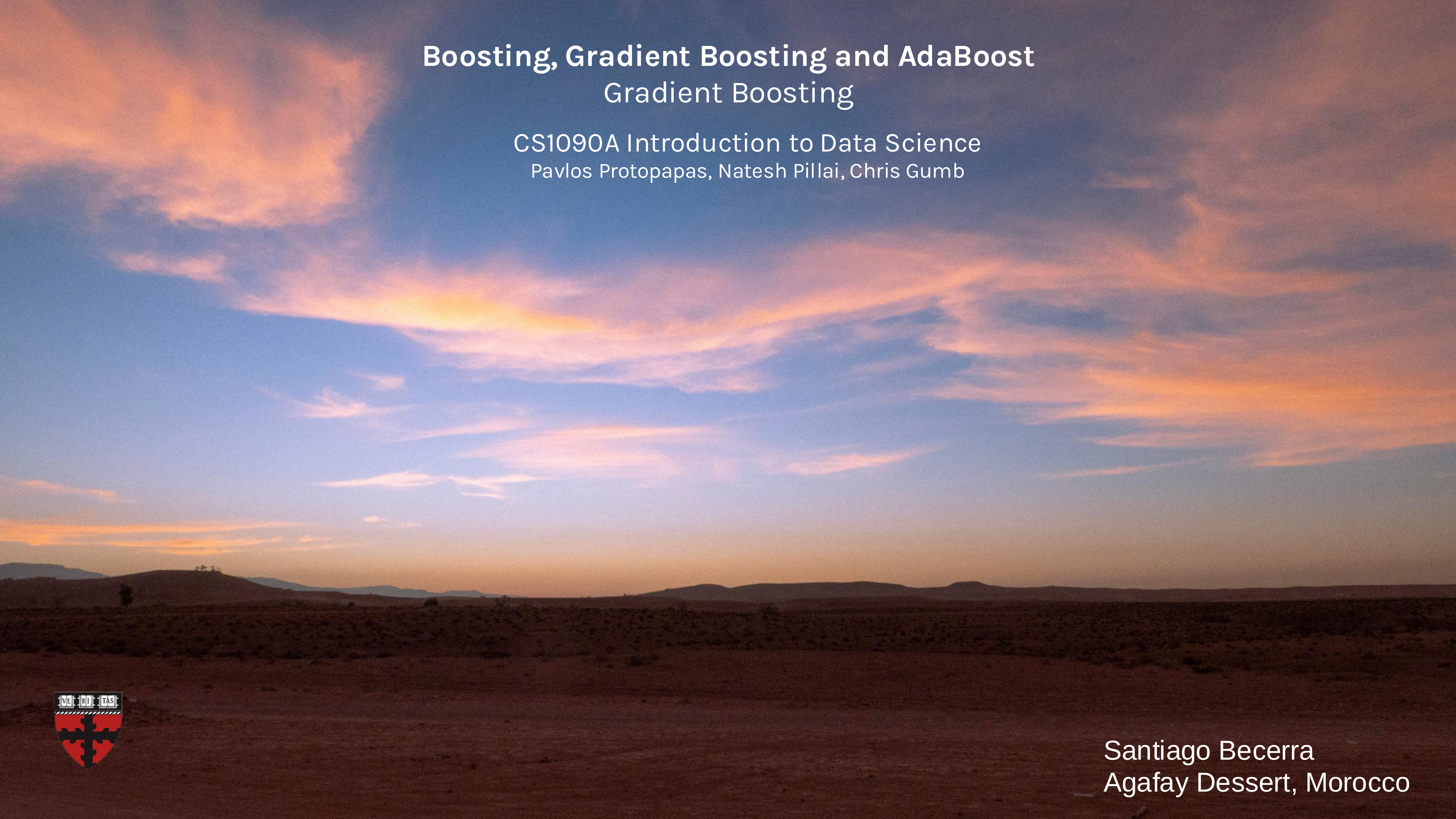
Boosting, Gradient Boosting and AdaBoost

Gradient Boosting

CS1090A Introduction to Data Science

Pavlos Protopapas, Natesh Pillai, Chris Gumb



Santiago Becerra
Agafay Dessert, Morocco

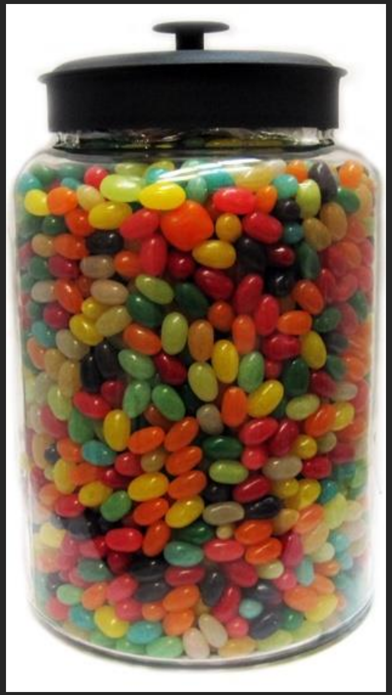
Outline

- Introduction to Boosting
- **Gradient Boosting**
- Mathematical Formulation - Gradient Boosting

Recap: Boosting

"Can a set of weak learners create a single strong learner?"

Leslie Gabriel Valiant



How many jellybeans do you see?

Recap: Boosting

The key intuition behind boosting is that one can take an ensemble of simple models $\{T_h\}_{h \in H}$ and additively combine them into a single, more complex model. Here, h is a weak learner and H is the hypothesis space of all possible weak learners.

Each model T_h might be a poor fit for the data, but a linear combination of the ensemble

$$\boxed{T = \sum_h \lambda_h T_h}$$

can be expressive and flexible.

Question: But which models should we include in our ensemble? What should the coefficients or weights in that linear combination be?

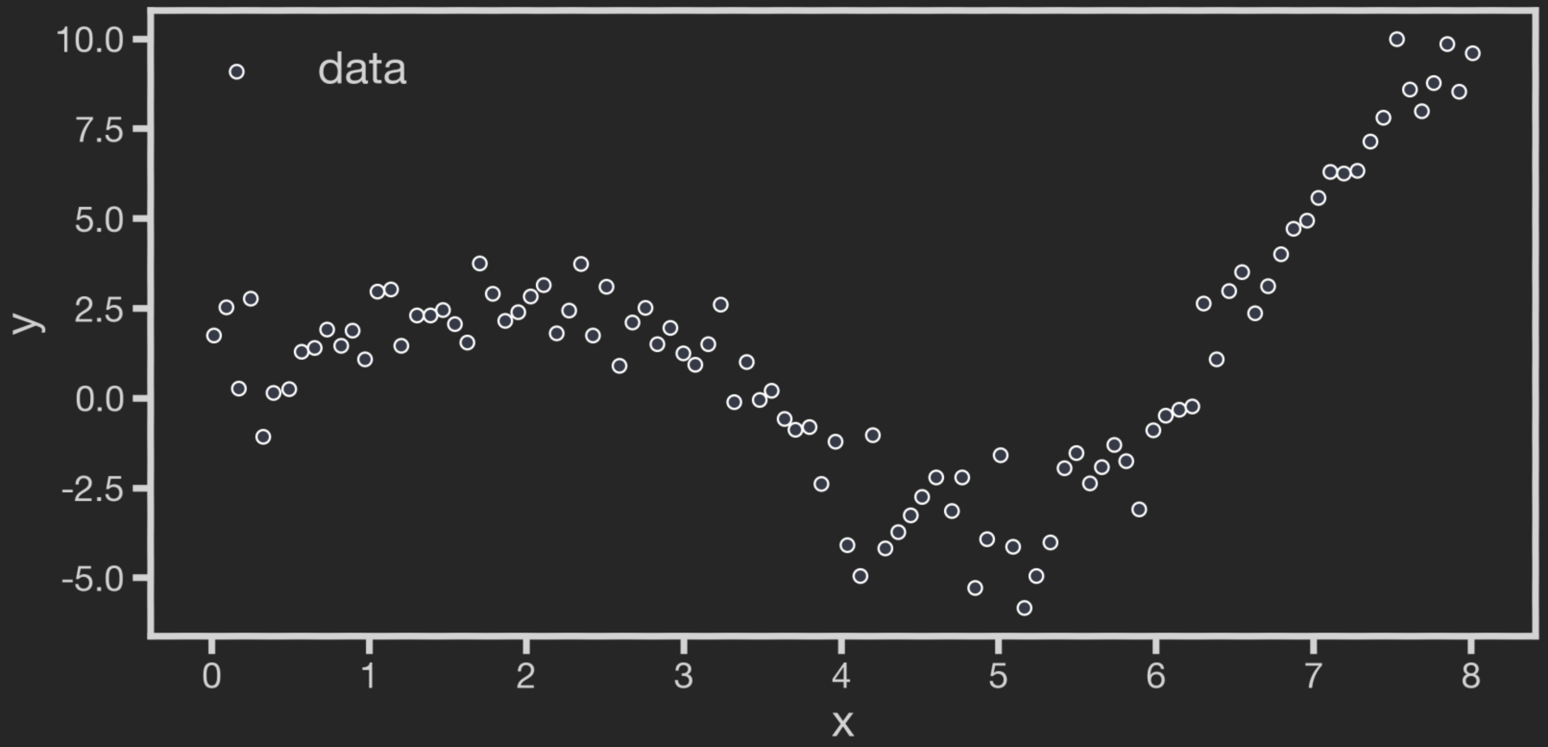
Gradient Boosting

There are two main ideas in gradient boosting:

- Gradient boosting is a method for iteratively building a complex model T by adding simple models.
- Each new simple model added to the ensemble compensates for the weaknesses of the current ensemble.

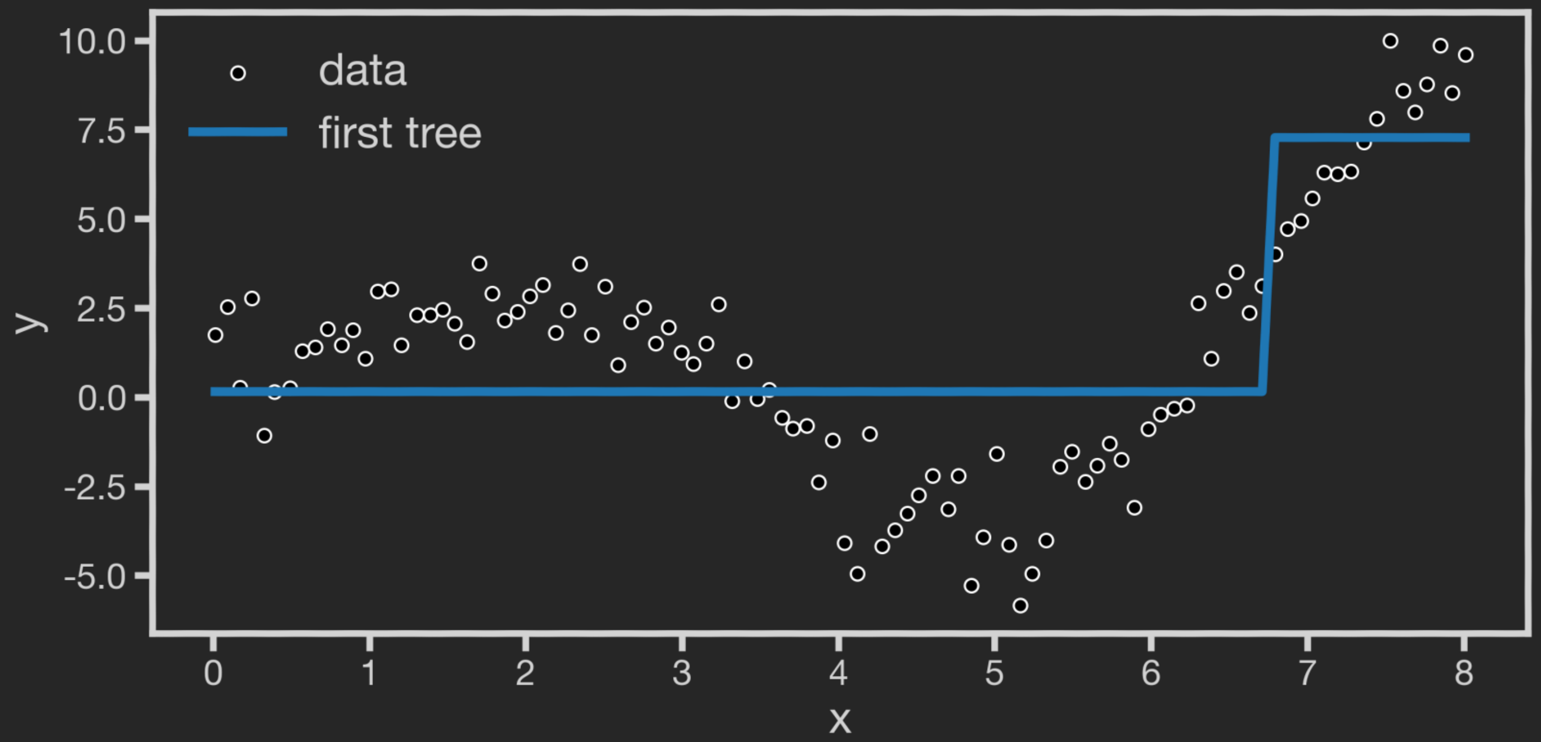
Gradient Boosting: Illustration

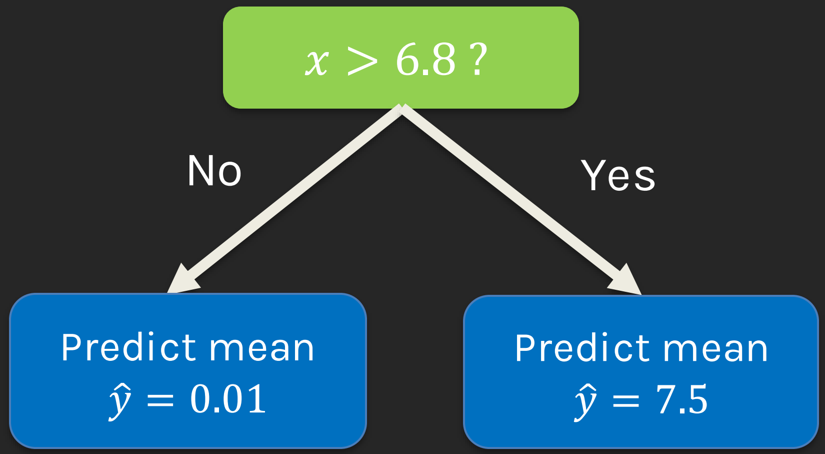
Consider the following dataset:



Gradient Boosting: Illustration

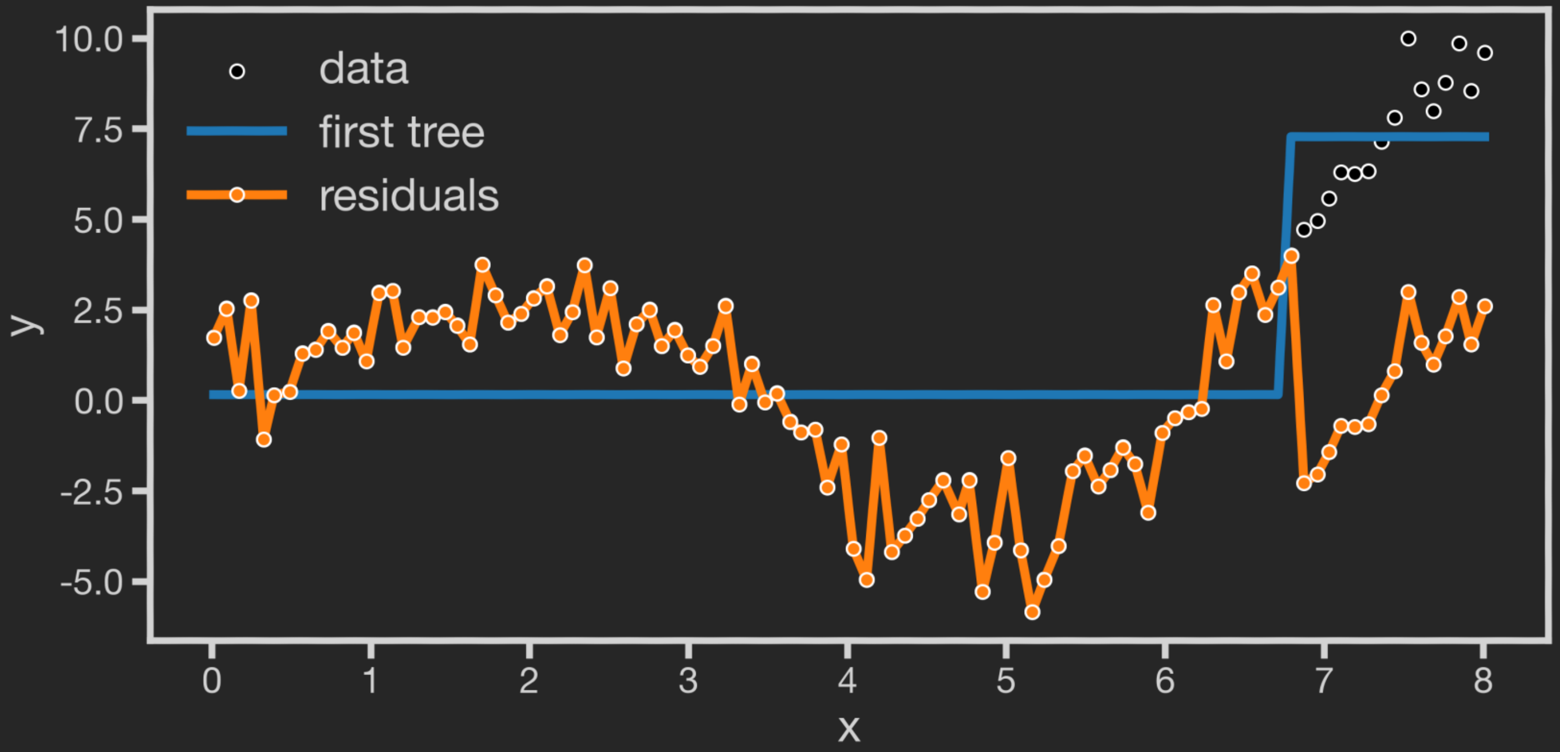
Step 1: Fit a simple model $T^{(0)}$ on the training data: $\{(x_1, y_1), \ldots, (x_N, y_N)\}$.

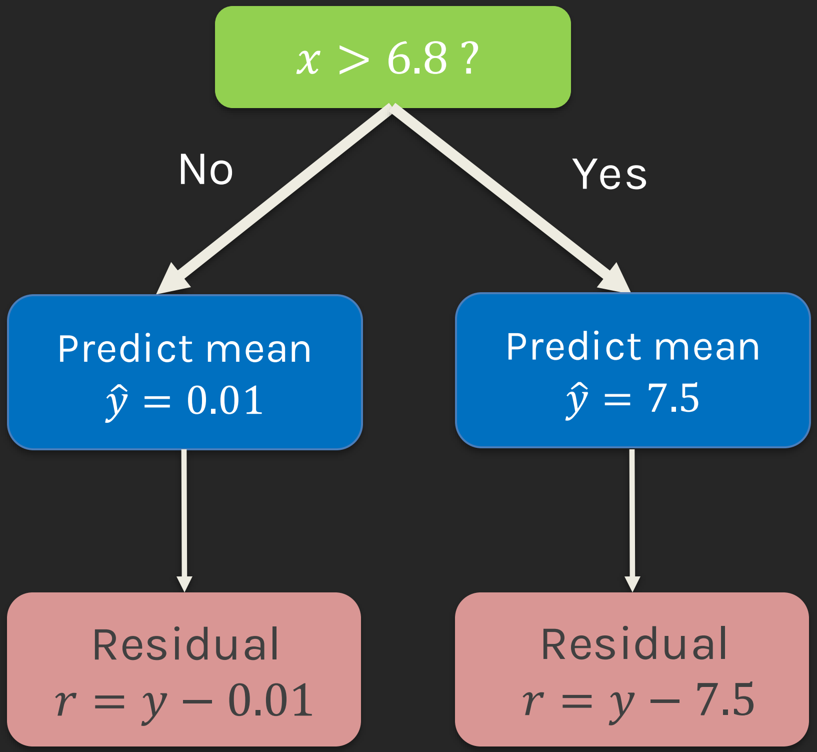




Gradient Boosting: Illustration

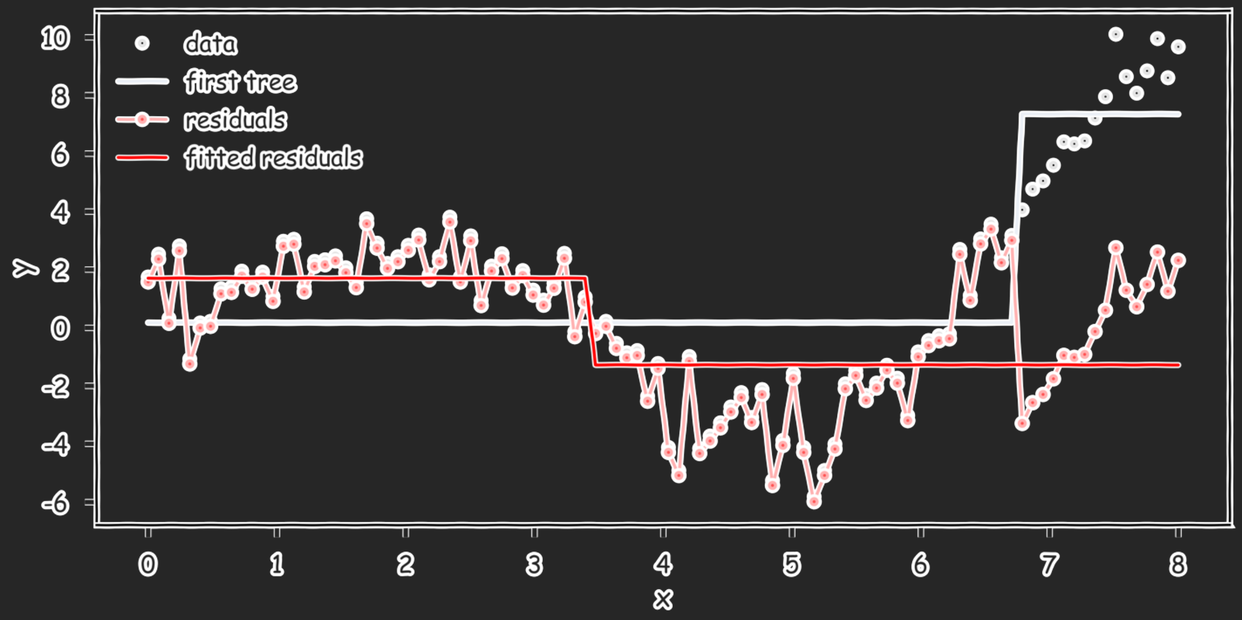
Step 2: Compute the residuals $\{r_1, \ldots, r_N\}$ for $T^{(0)}$. Set $T \leftarrow T^{(0)}$.

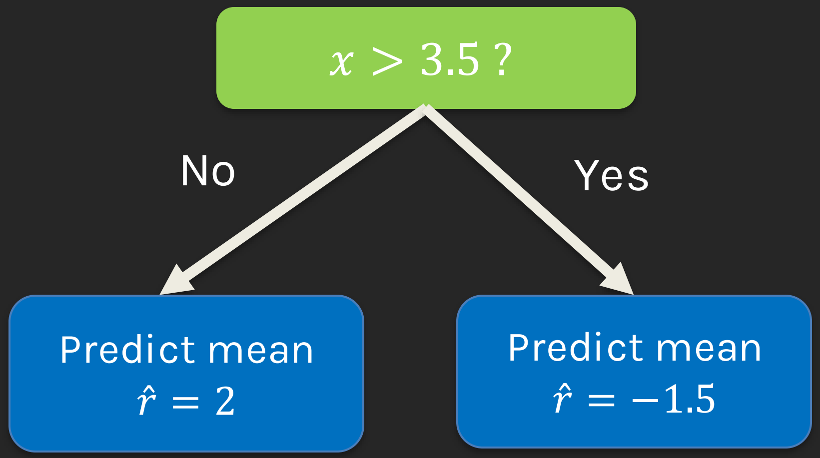




Gradient Boosting: Illustration

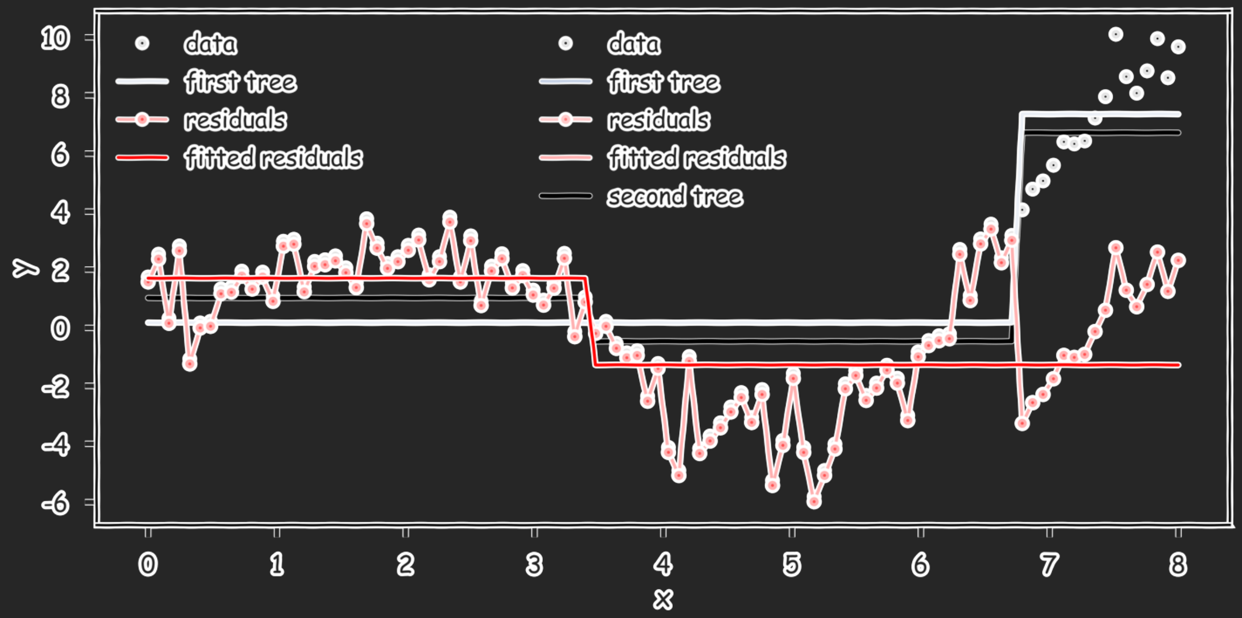
Step 3: Fit another model $T^{(1)}$ on: $\{(x_1, r_1), \ldots, (x_N, r_N)\}$.

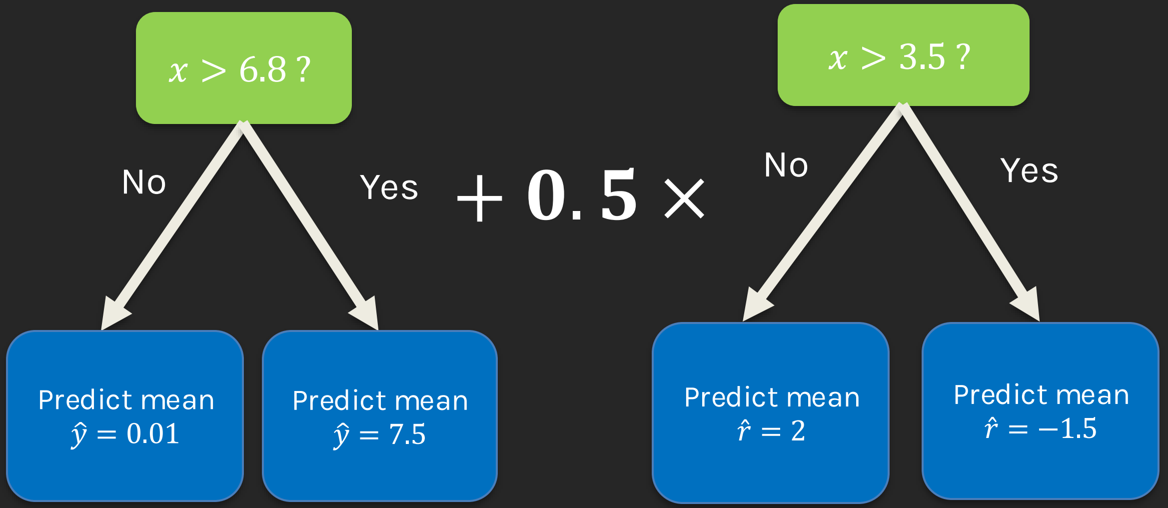




Gradient Boosting: Illustration

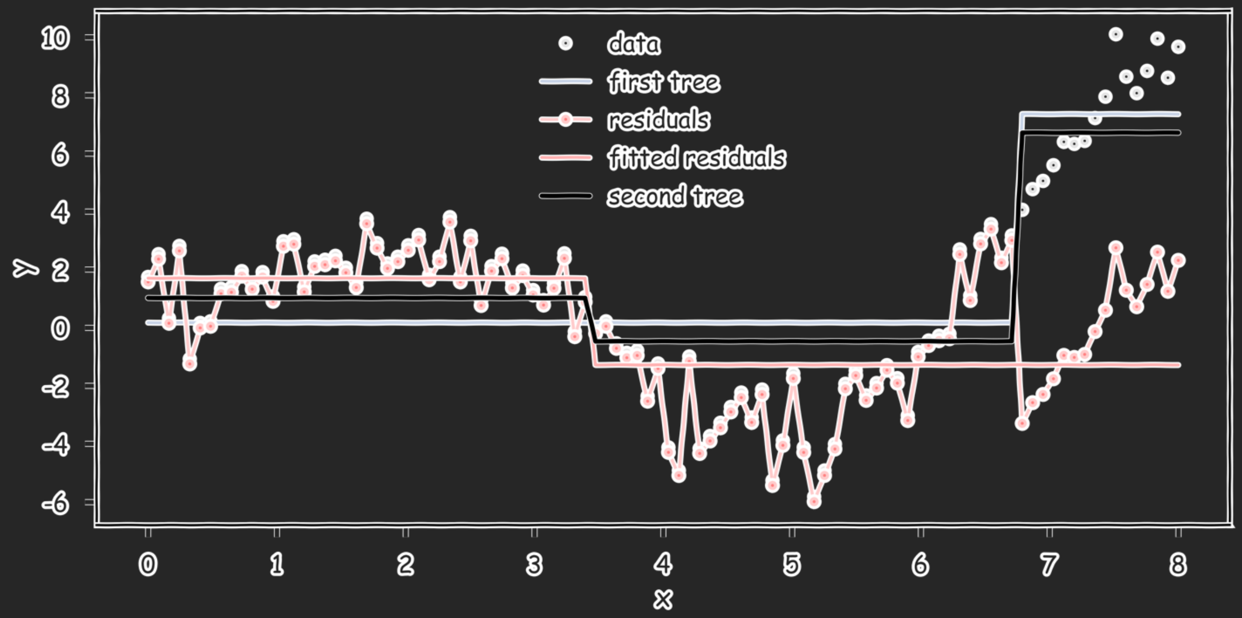
Step 4: Combine the two trees in step 1 and 3 by setting $T \leftarrow T + \lambda T^{(1)}$.
Assume $\lambda = 0.5$.

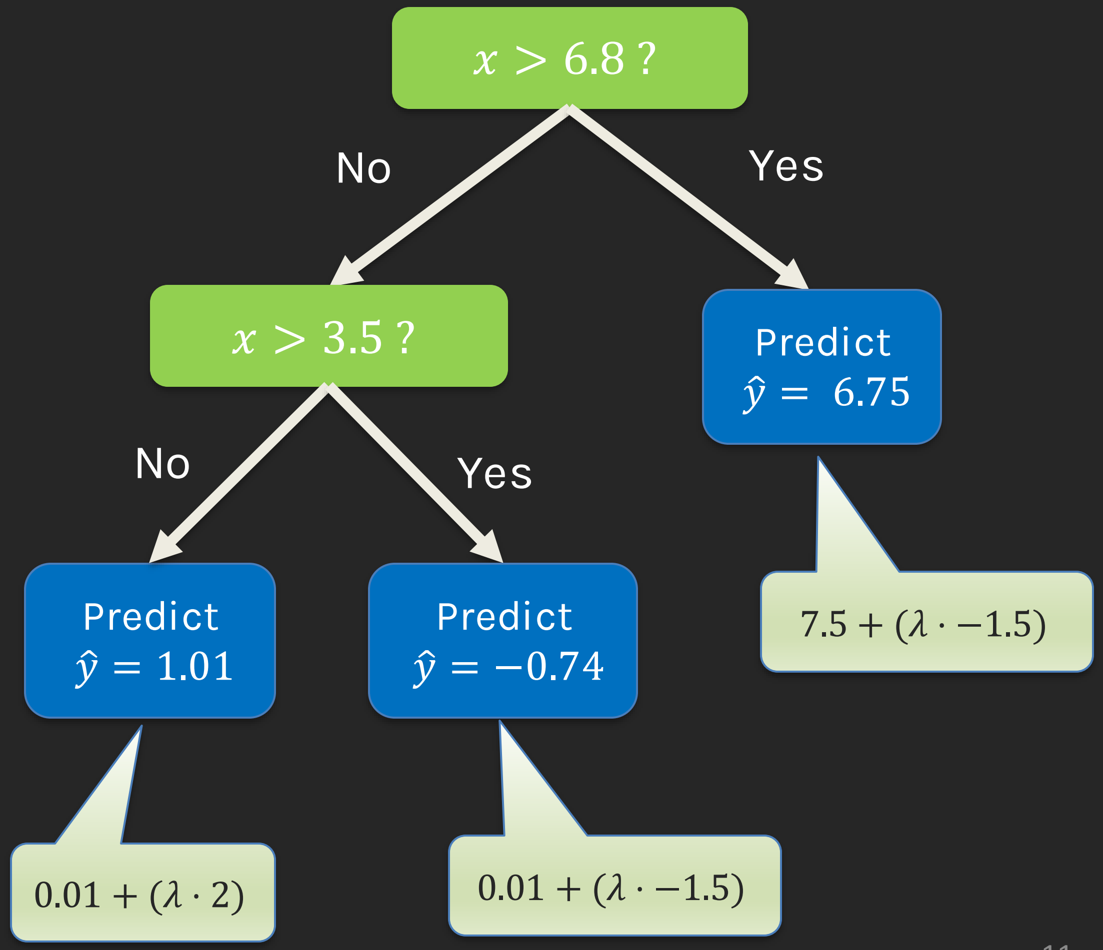




Gradient Boosting: Illustration

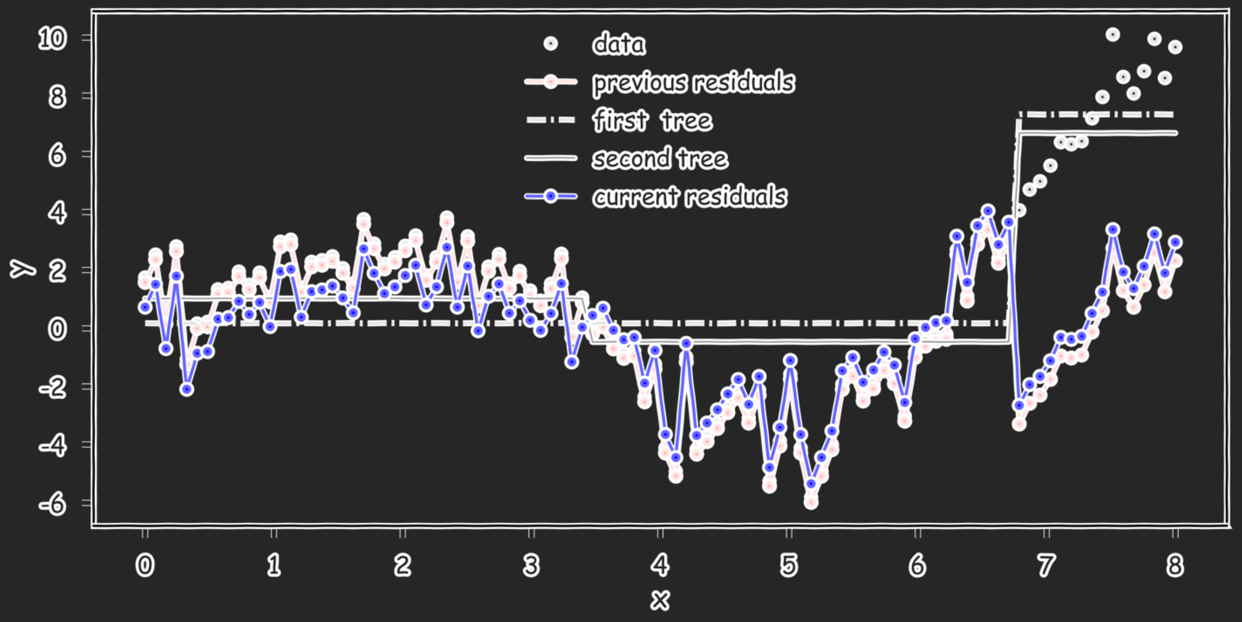
Step 4: Combine the two trees in step 1 and 3 by setting $T \leftarrow T + \lambda T^{(1)}$.
Assume $\lambda = 0.5$.

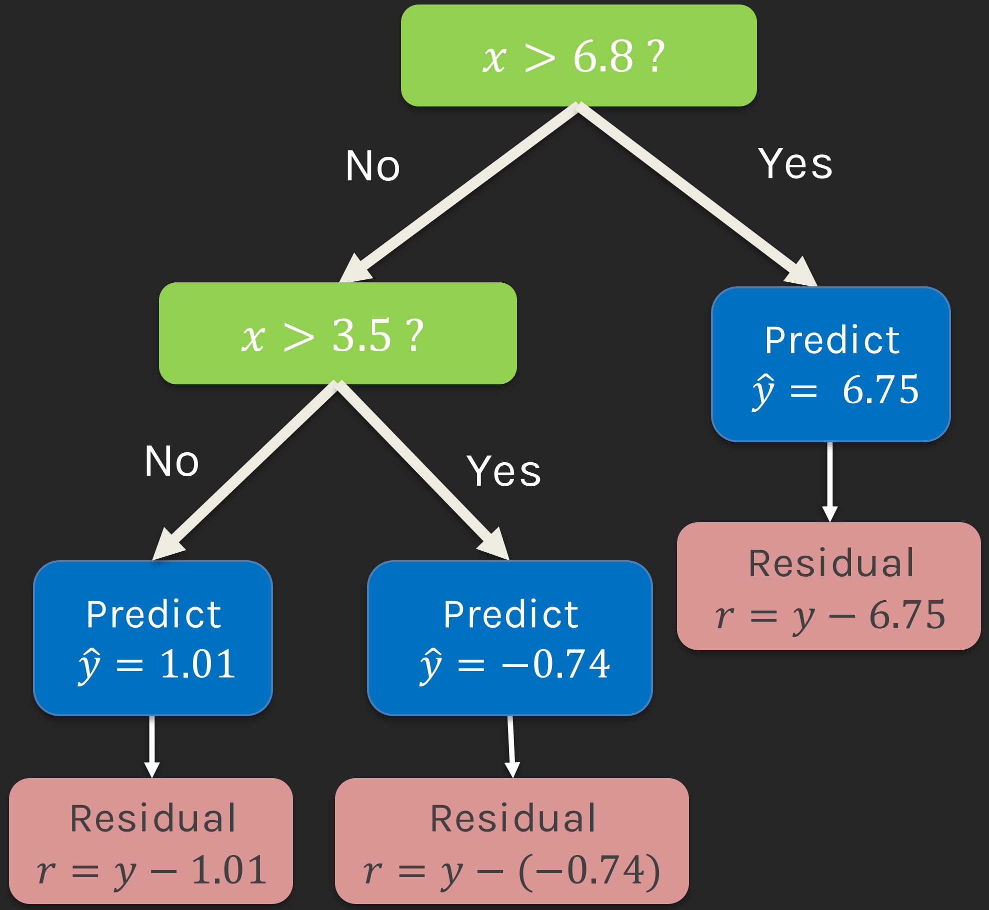




Gradient Boosting: Illustration

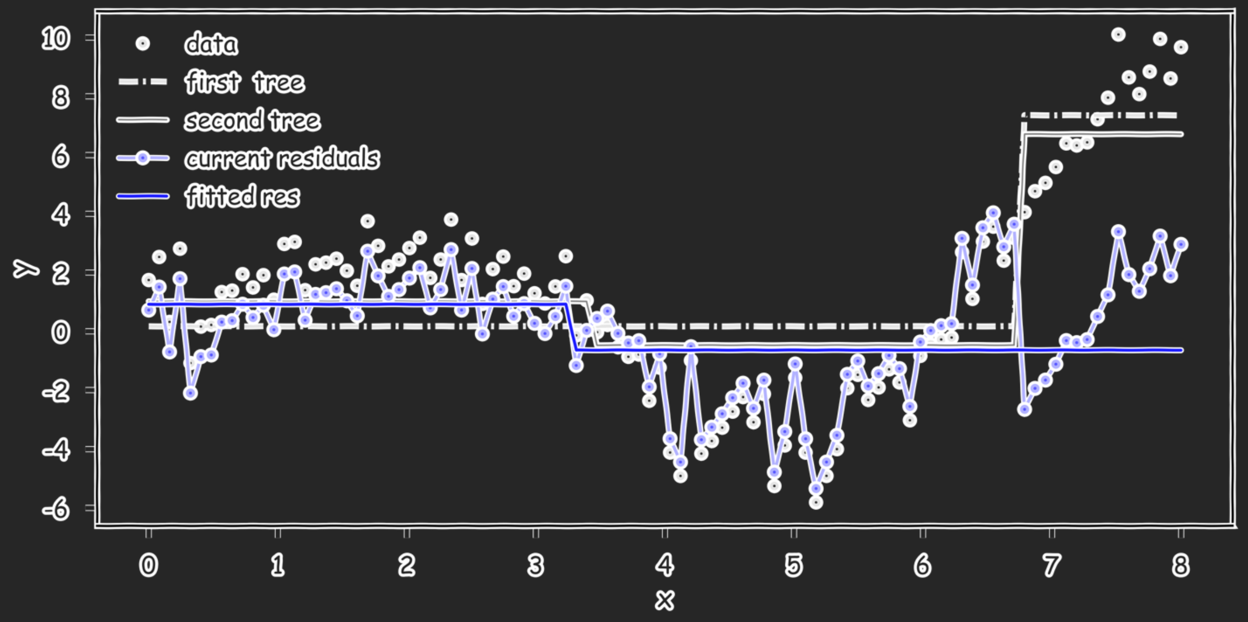
Step 5: Repeat step 2 on the new model by calculating the residuals $\{r_1, \ldots, r_N\}$ for current model T.

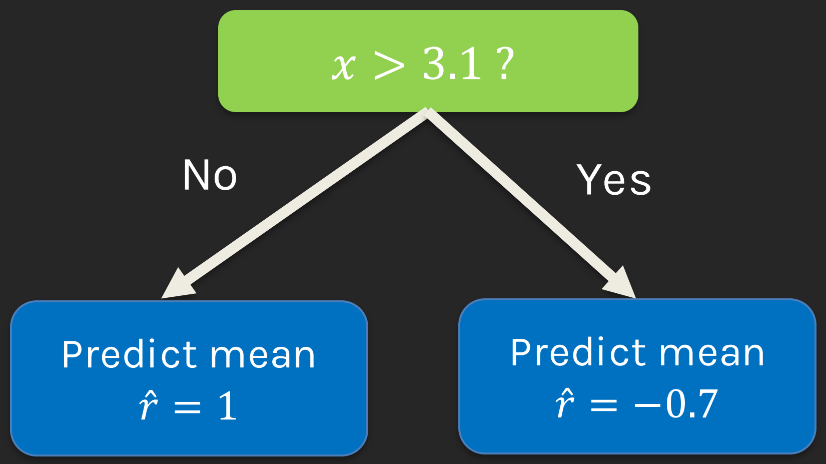




Gradient Boosting: Illustration

Step 6: Repeat step 3 and fit another model $T^{(2)}$ on the new residuals: $\{(x_1, r_1), \ldots, (x_N, r_N)\}$.

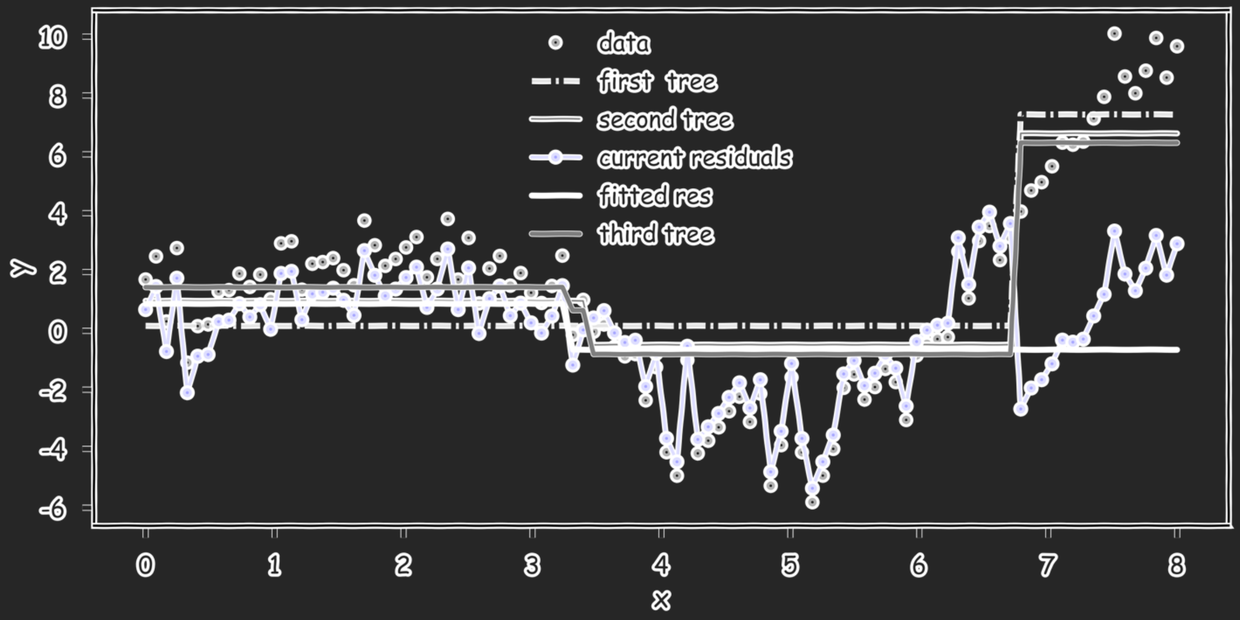


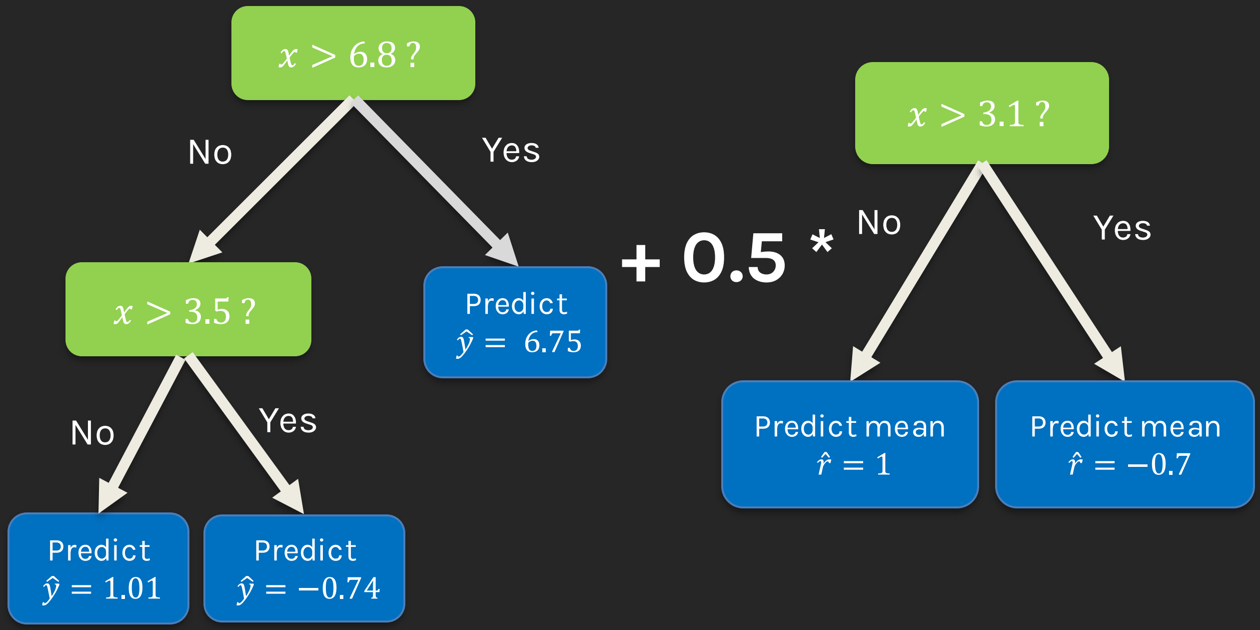


Gradient Boosting: Illustration

Step 7: Combine the two trees in step 4 and 6 by setting $T \leftarrow T + \lambda T^{(2)}$.

Assume $\lambda = 0.5$.

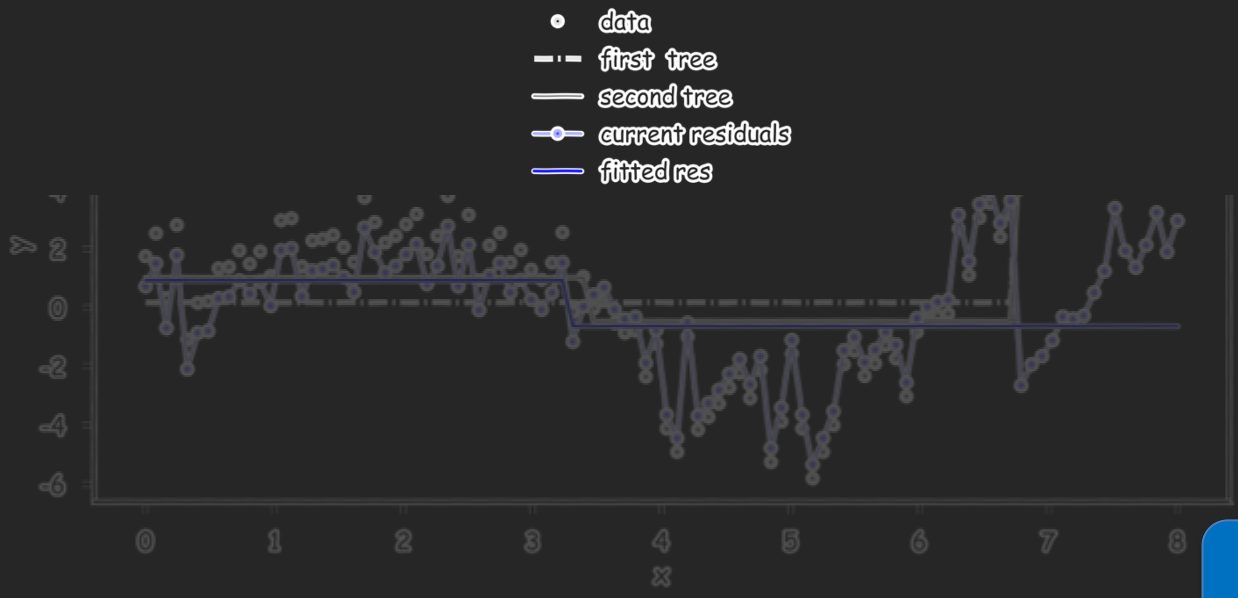


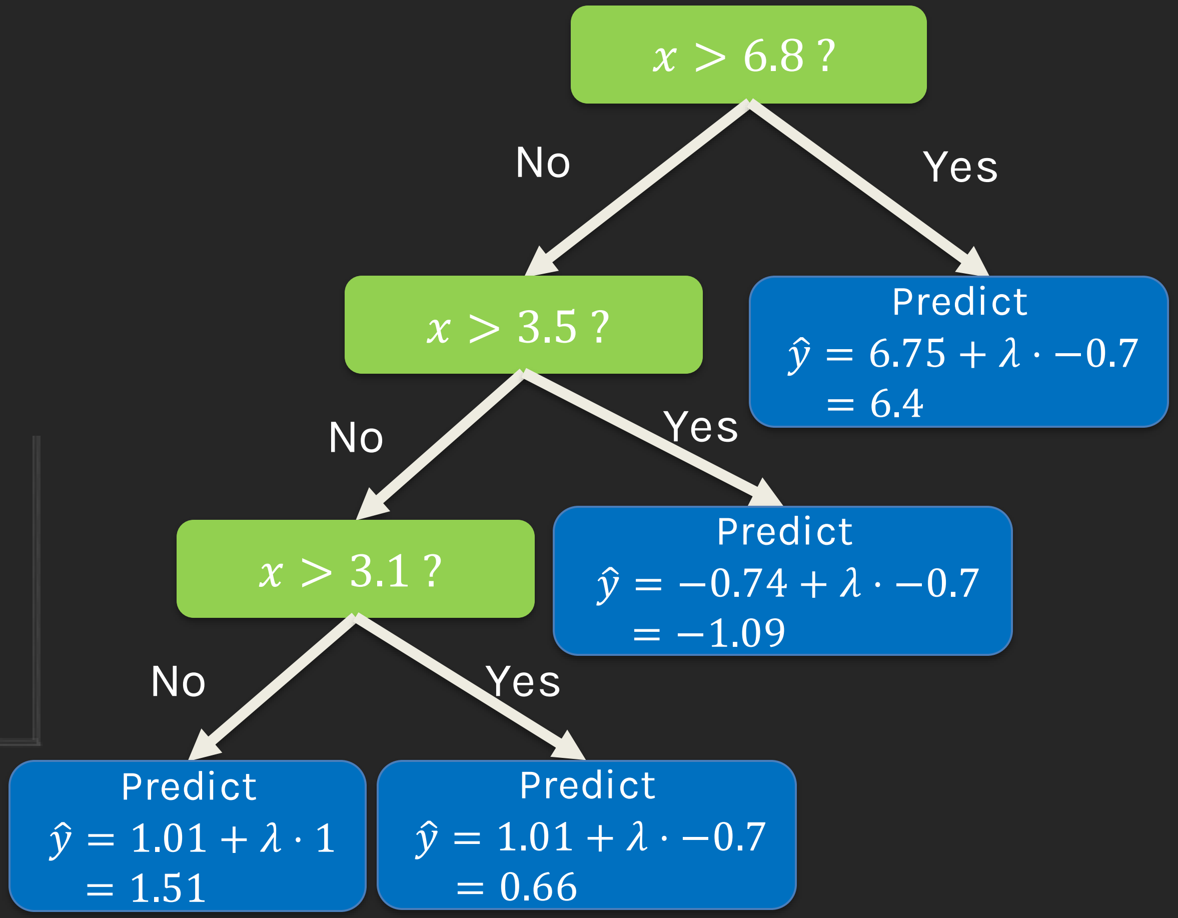


Gradient Boosting: Illustration

Step 7: Combine the two trees in step 4 and 6 by setting $T \leftarrow T + \lambda T^{(2)}$.

Assume $\lambda = 0.5$.





Gradient Boosting: Algorithm

Step 1: Fit a simple model $T^{(0)}$ on the training data $\{(x_1, y_1), \ldots, (x_N, y_N)\}$. Set $T \leftarrow T^{(0)}$.

Step 2: Compute the residuals $\{r_1, \ldots, r_N\}$ for T.

For $i = 1 \ldots$ until stopping condition is met:

Step 3: Fit a simple model, $T^{(i)}$, to the current **residuals**.

That is, train $T^{(i)}$ using $\{(x_1, r_1), \ldots, (x_N, r_N)\}$

Step 4: Set the current model $T \leftarrow T + \lambda T^{(i)}$.

Step 5: Compute residuals at step i, set $r_n \leftarrow r_n - \lambda T^{(i)}(x_n),\ n = 1, \ldots, N$

where λ is a constant called the **learning rate**.