Boosting Techniques

Understanding Boosting Algorithms for Machine Learning

Janani Ravi

Co-founder, Loonycorn

www.loonycorn.com

System and Software Requirements

Windows, MacOS, or Linux Machine

Recent version of Python v3.8+

System and Software Requirements

- scikit-learn 1.6.1
- xgboost 3.0.0
- lightgbm 4.6.0
- catboost 1.23.5

Ensemble Learning Techniques and Concepts

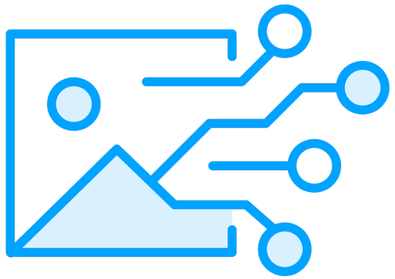
Ensemble Learning

Machine learning technique in which several learners are combined to obtain a better performance than any of the learners individually.

Benefits of Ensemble Learning

Better accuracy

Reduced risk of overfitting



Better generalization

Robustness to noise and outliers

Important Questions in Ensemble Learning

What kind of individual learners to use?

How should individual learners be trained?

How should individual learners be combined?

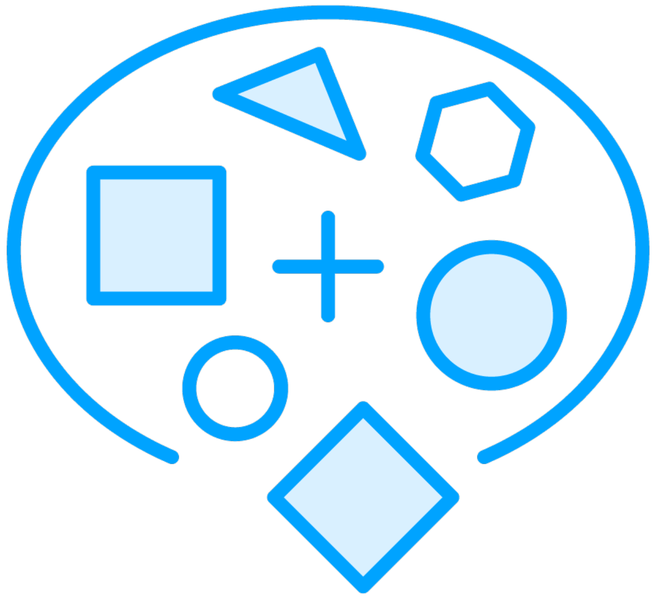
Important Questions in Ensemble Learning

What kind of individual learners to use?

How should individual learners be trained?

How should individual learners be combined?

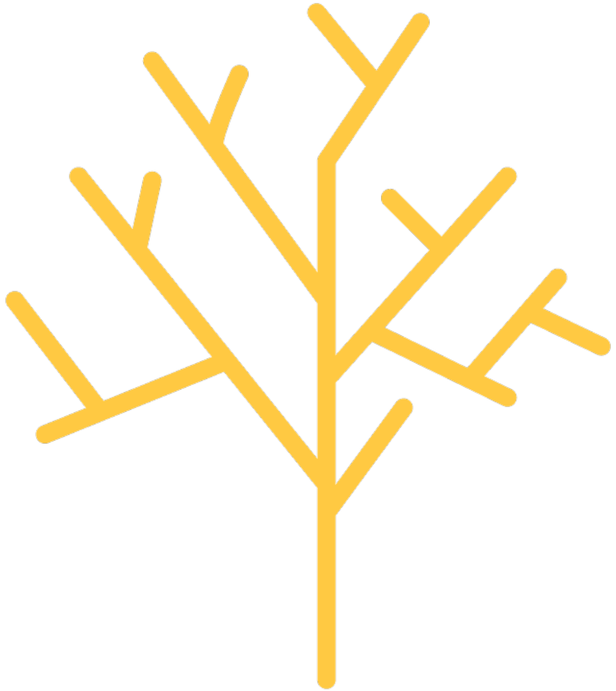
Choice of Individual Learners



Individual learners (models) could be of absolutely any type

Each learner should be as **different** as possible from other learners

Choice of Individual Learners



Each learner is very simple (weak) on its own

Typically a very shallow decision tree

Combined, the learners cover up each others' faults

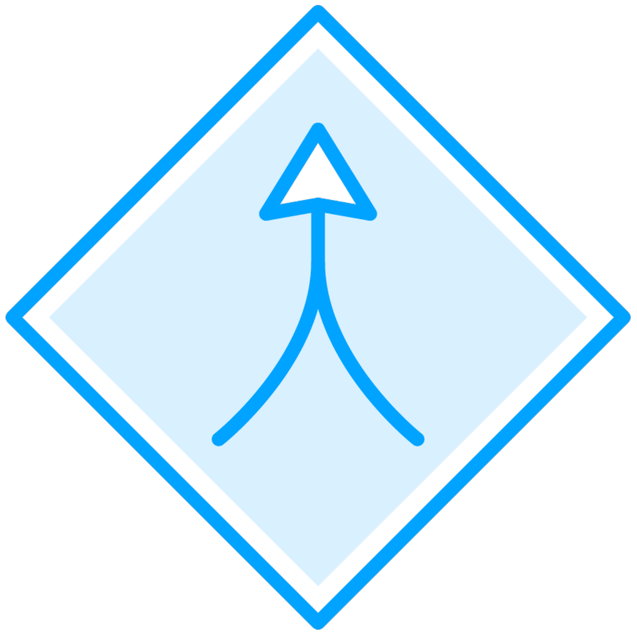
Important Questions in Ensemble Learning

What kind of individual learners to use?

How should individual learners be trained?

How should individual learners be combined?

Combining Individual Learners



Averaging

Voting

Stacking

Important Questions in Ensemble Learning

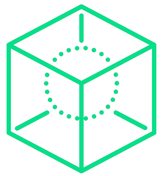
What kind of individual learners to use?

How should individual learners be trained?

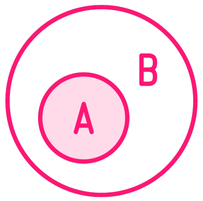
How should individual learners be combined?

The Learners Should Be Diverse

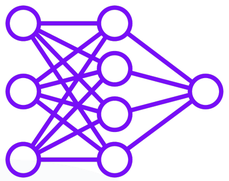
Individual models should make different types of errors



Same model + same data = same mistakes



Option 1: different data subsets



Option 2: different types of models

Similar Learners = Same Learning from Data

Different Learners = Diverse Learning from Data

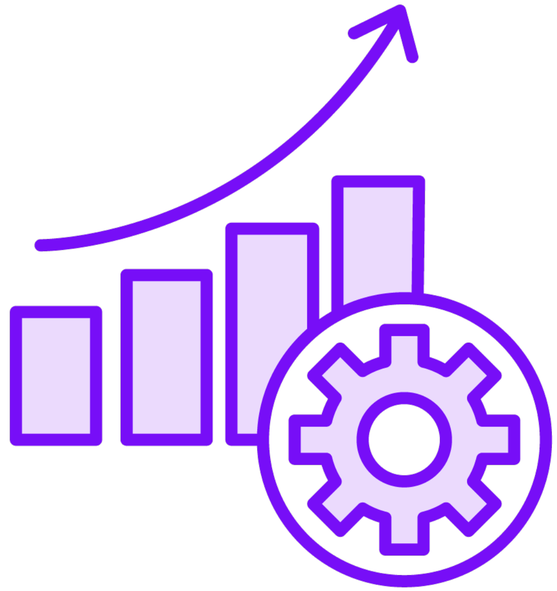
Training Ensemble Models

Training Ensemble Models

Averaging vs. Boosting

Averaging

Train predictors in parallel and average scores of individual predictors



Boosting

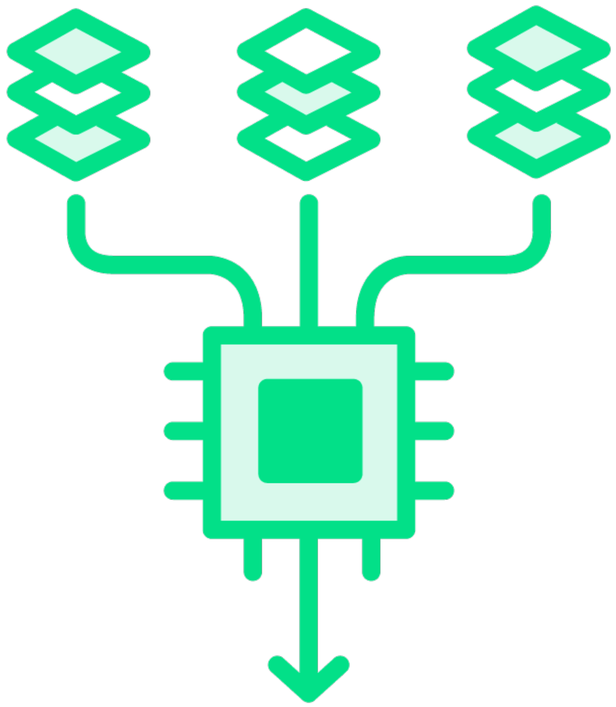
Train predictors in sequence where each predictor learns from earlier mistakes

Averaging vs. Boosting

Averaging	VS.	Boosting
Individual learners are independent		Individual learners are linked to previous learners
Can build trees in parallel		Need to build tree sequentially
Learners do not learn from mistakes of other learners		Individual learners explicitly configured to learn from previous mistakes

Training Ensemble Models

Averaging



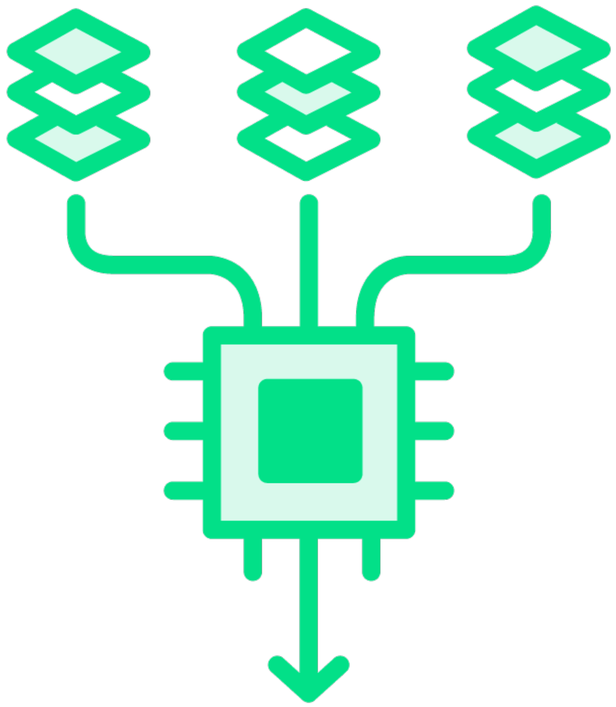
Train multiple learners in parallel

Get individual predictions from each learner

Final prediction of the ensemble is an average of individual predictions

Voting can be considered an averaging technique

Averaging

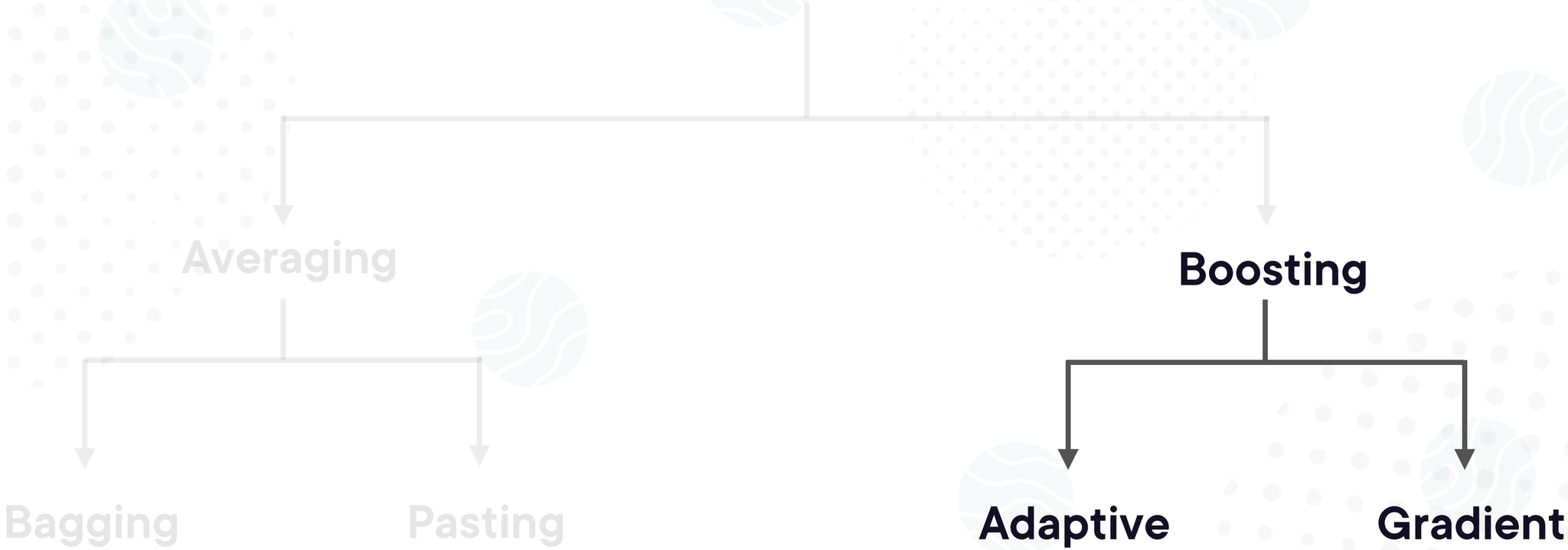


Usually use decision trees or random forests to build different models

Train model on different samples of training data

- Bagging: Sample data with replacement
- Pasting: Sample data without replacement

Training Ensemble Models



Boosting

Train multiple learners sequentially

Each model learns from the mistakes made by previous models

Can tweak the learning rate or contribution of each model

Addition of a learner boosts the accuracy of the model

Boosting

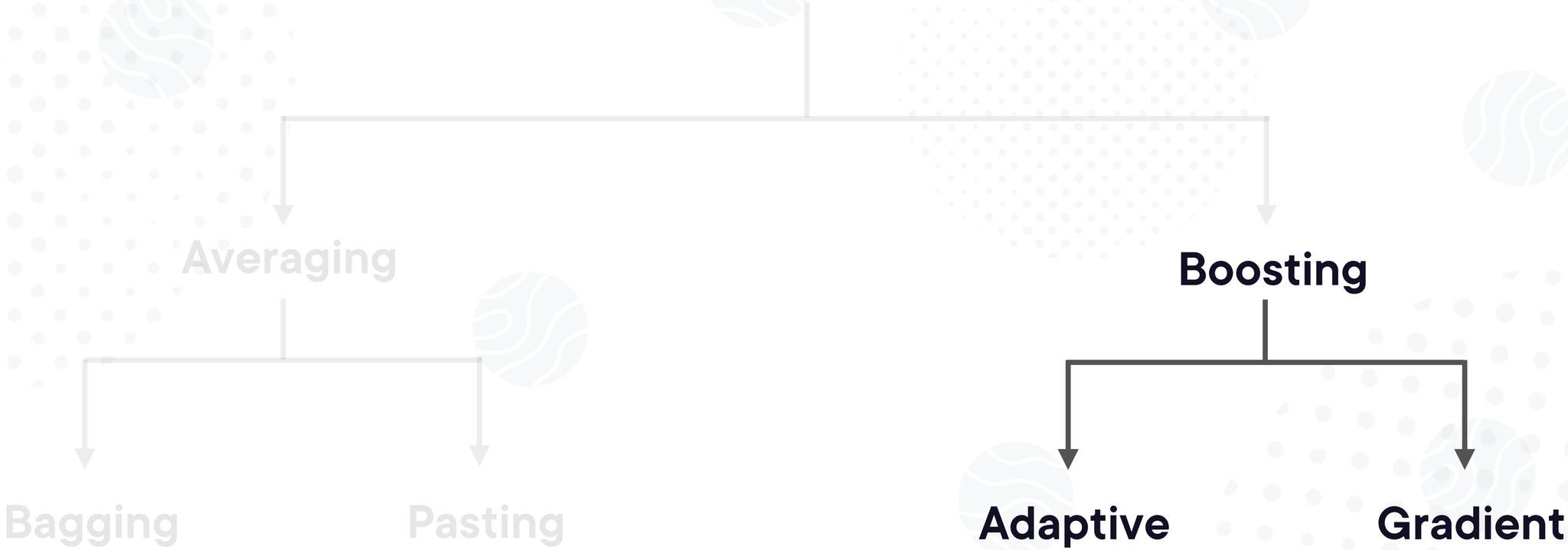
Adaptive Boosting: each model pays more attention to training instances the previous model got wrong

Gradient Boosting: each model in sequence fits on residual errors of the previous model

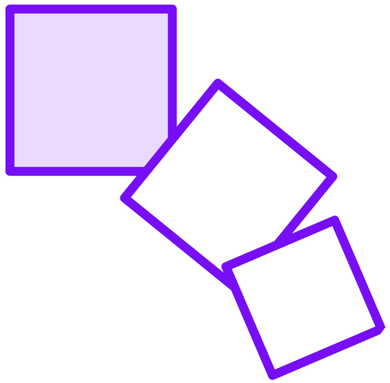
Boosting Techniques

Build up your weaknesses until they become your strengths.

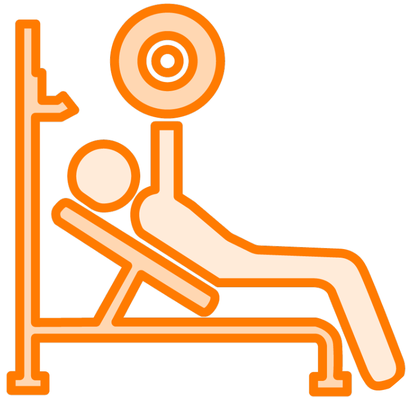
Training Ensemble Models



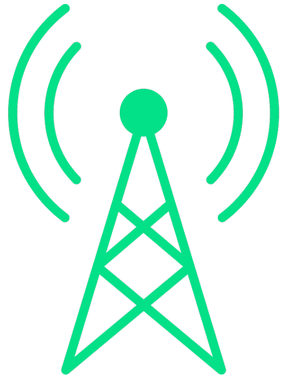
Adaptive Boosting (AdaBoost)



Models are built sequentially



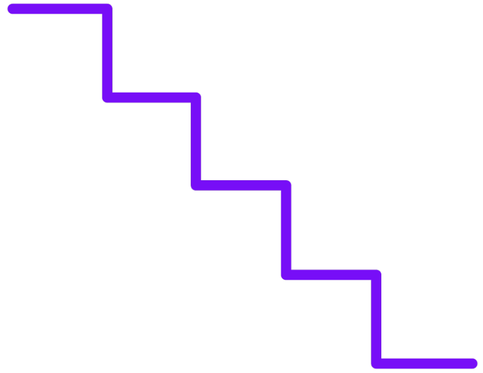
Assigns higher model weights to misclassified samples



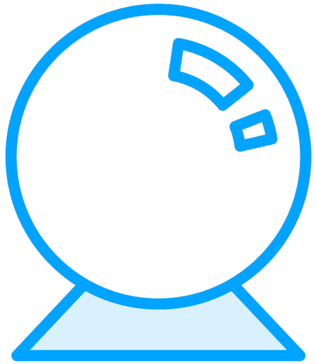
Each model's influence is based on accuracy

Model results combined for ensemble vote

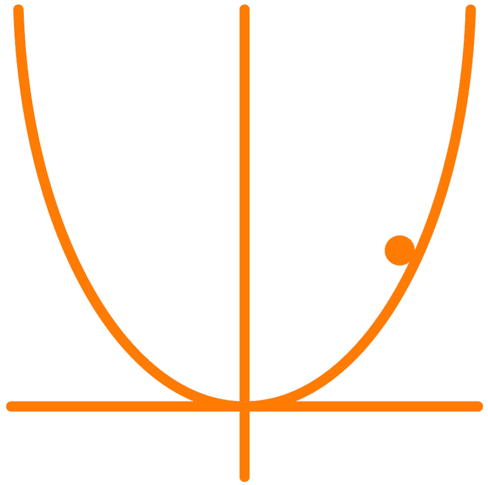
Gradient Boosting



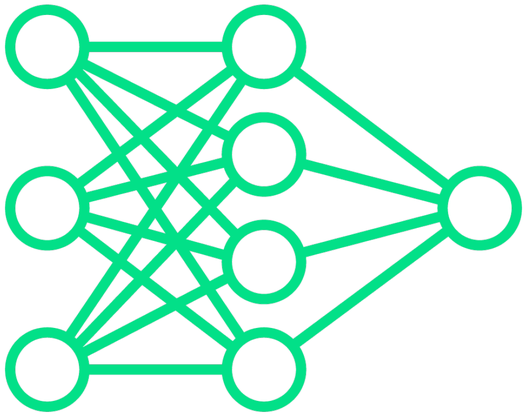
Models are trained sequentially



Each model predicts the errors of previous models



Uses gradients of loss functions



All model outputs combined

Benefits of Boosting Models

- High accuracy in predictions
- Focuses on improving where it struggled
- Balances bias and variance
- Works well with weak learners (shallow decision trees)

Drawbacks of Boosting Models

- Prone to overfitting
- Slower to train (models must be built sequentially)
- Sensitive to outlier samples
- Harder to tune

Adaptive Boosting (AdaBoost)

*Ada*ptive *Boost*ing (AdaBoost)

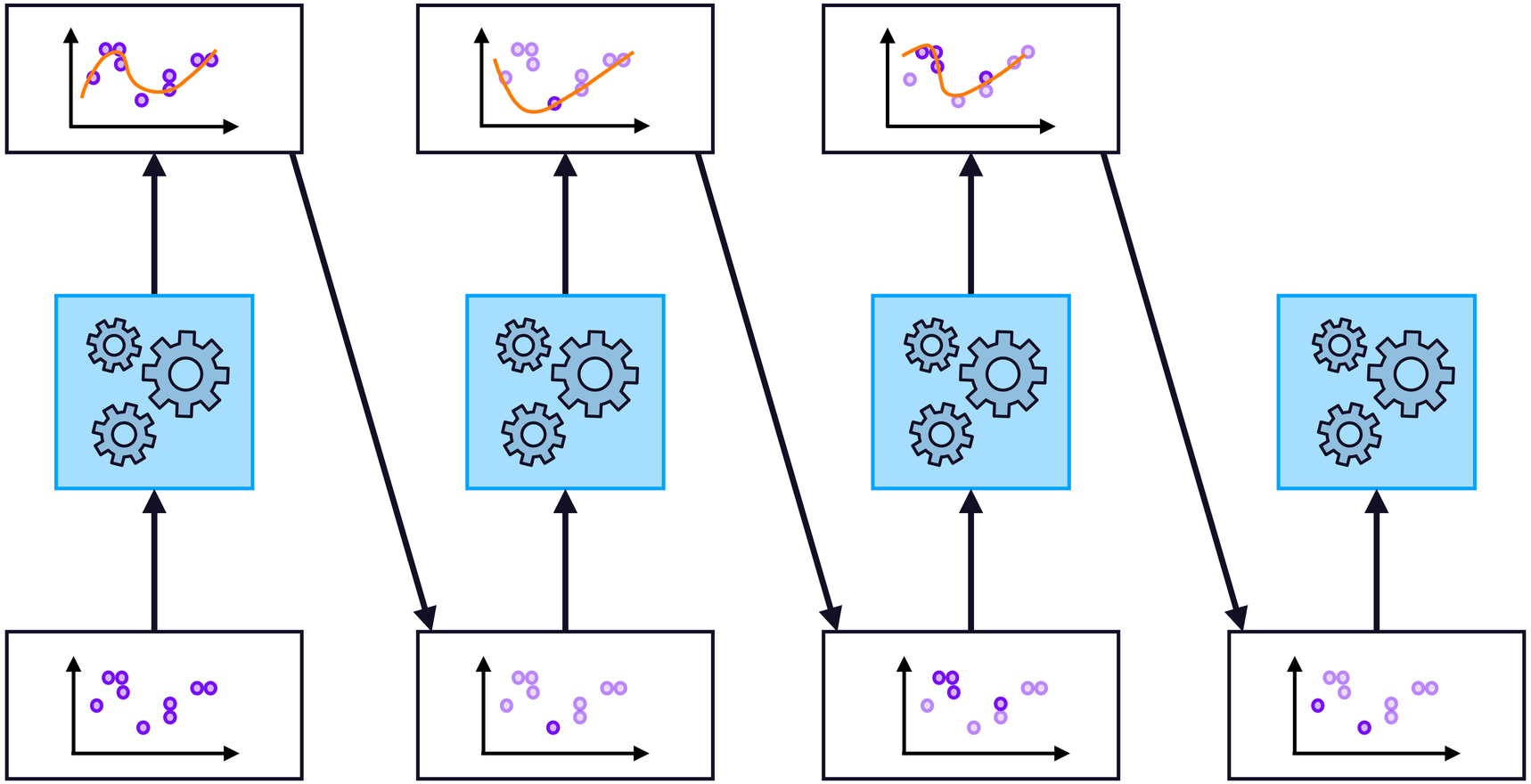
Construct and train models sequentially

Misclassified training points from each model are up-weighted

Next model sees previously misclassified points more often

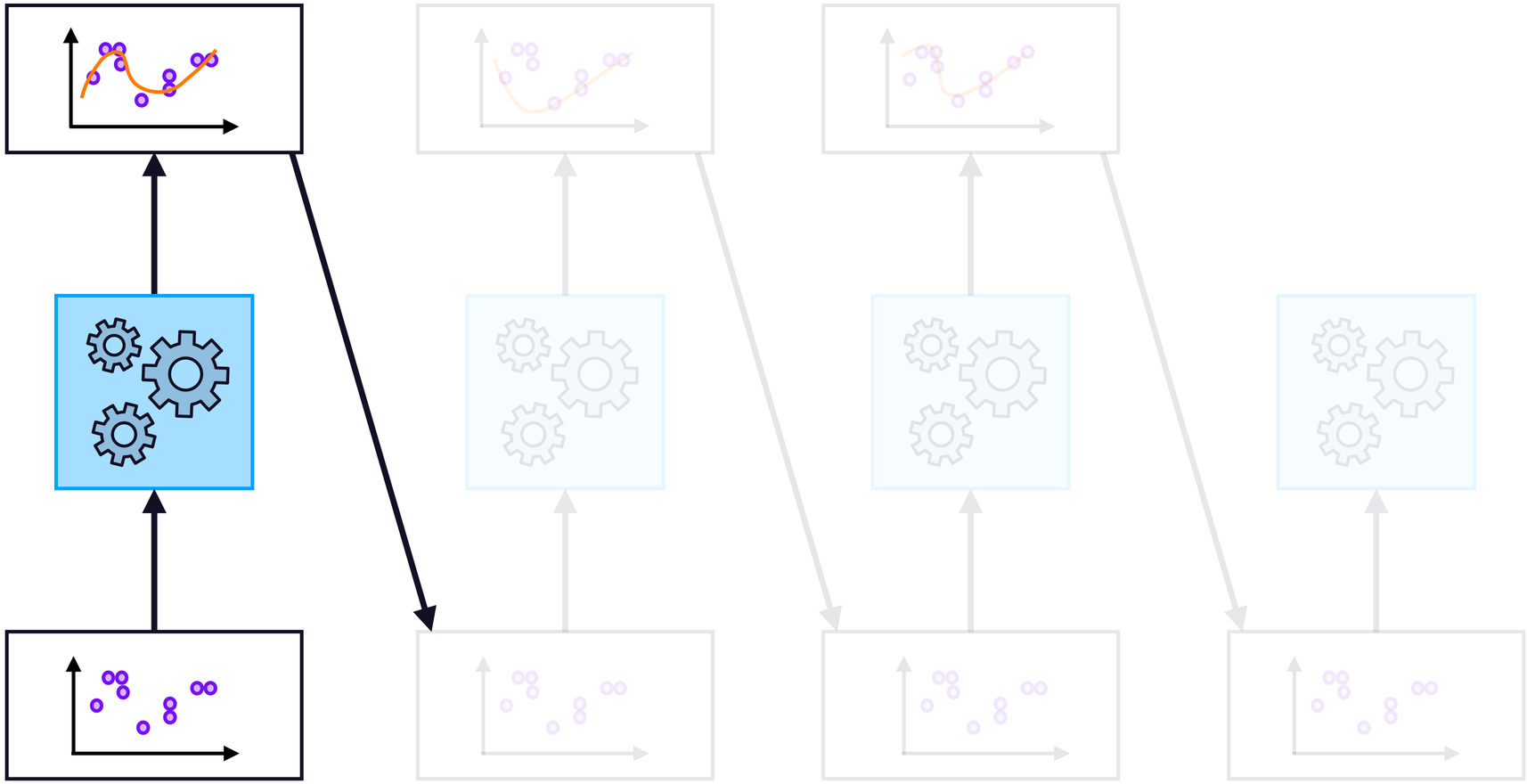
AdaBoost

Sequential training



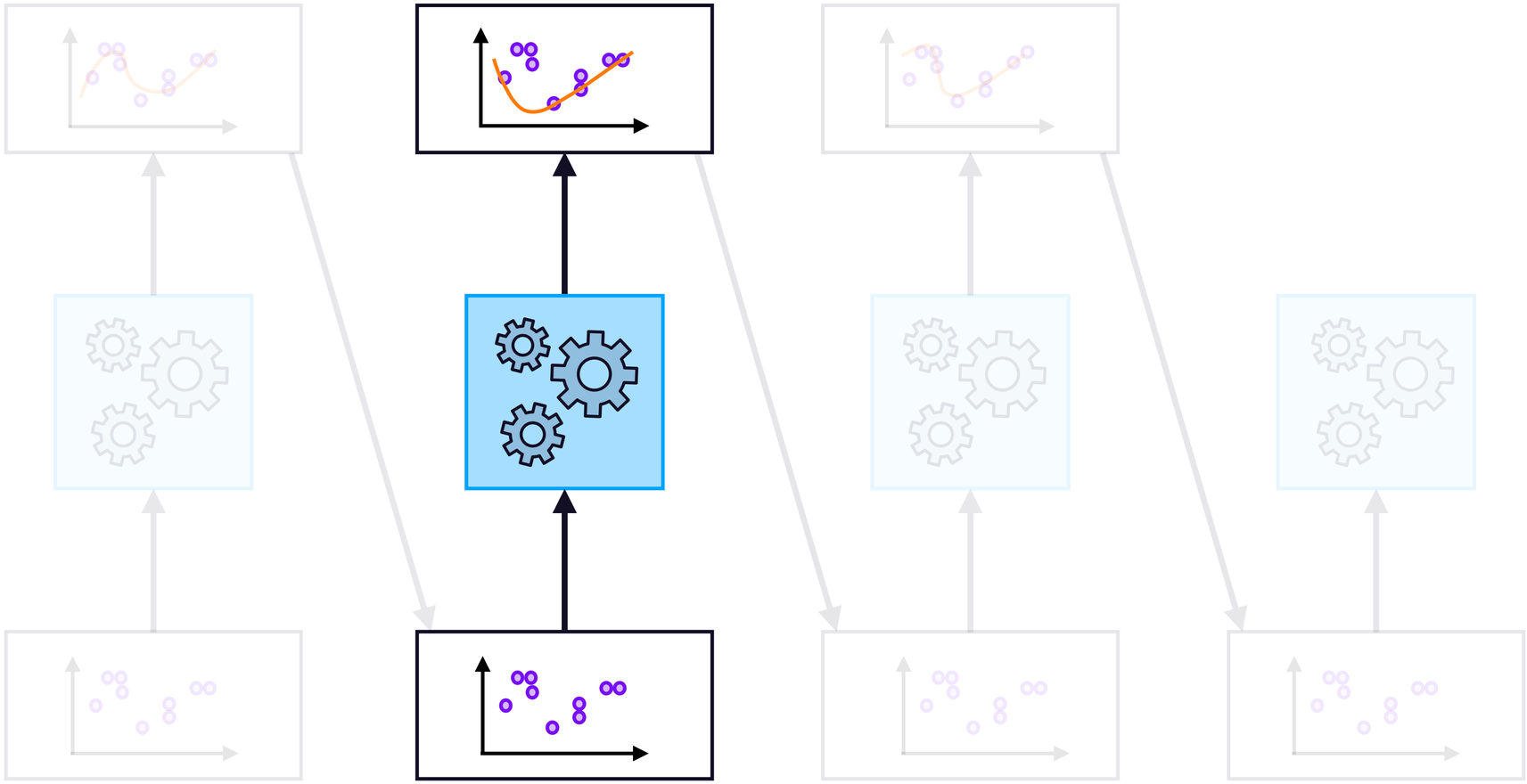
AdaBoost

Train the first predictor in a sequence



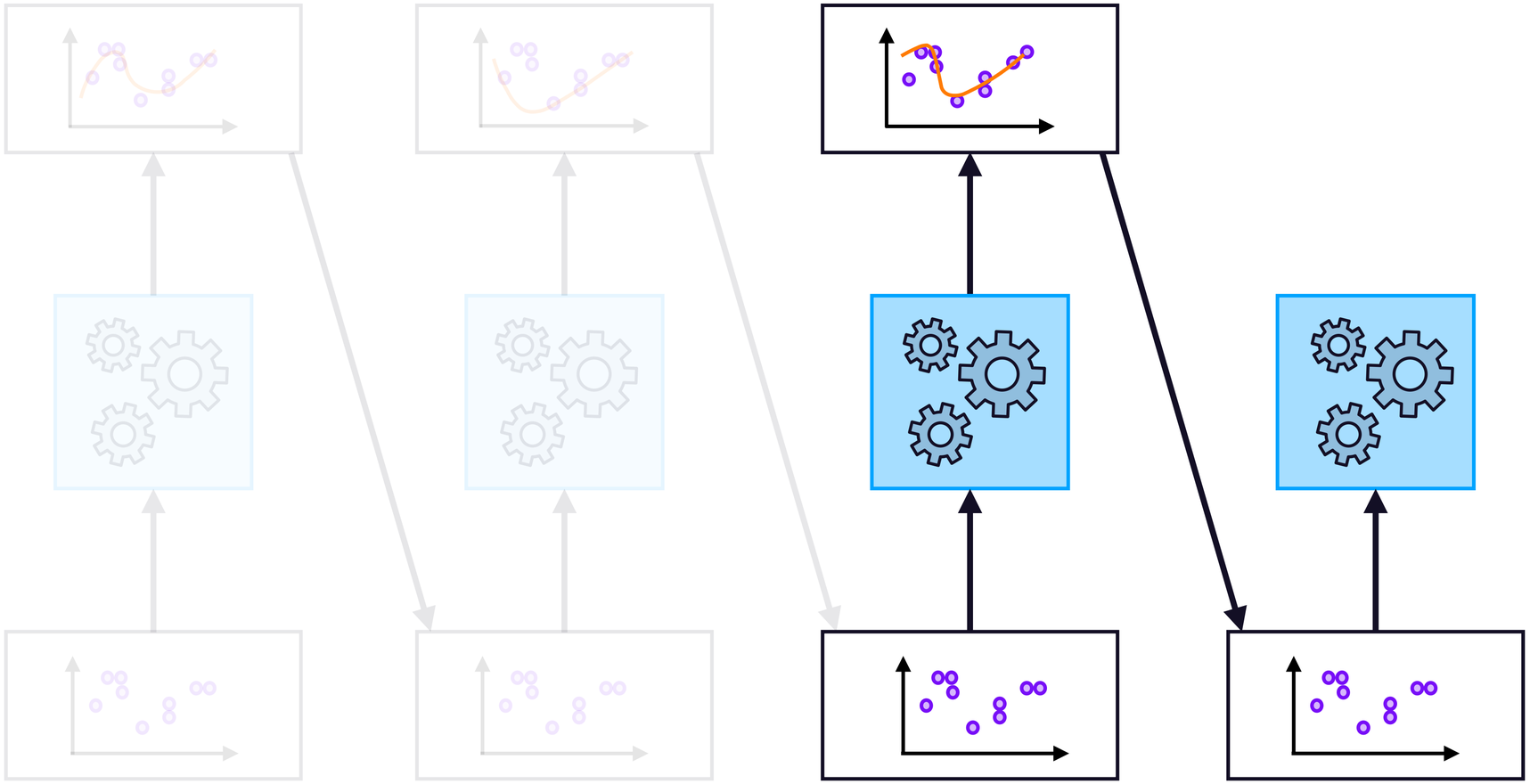
AdaBoost

Increase relative weights of those instances that previous predictors got wrong



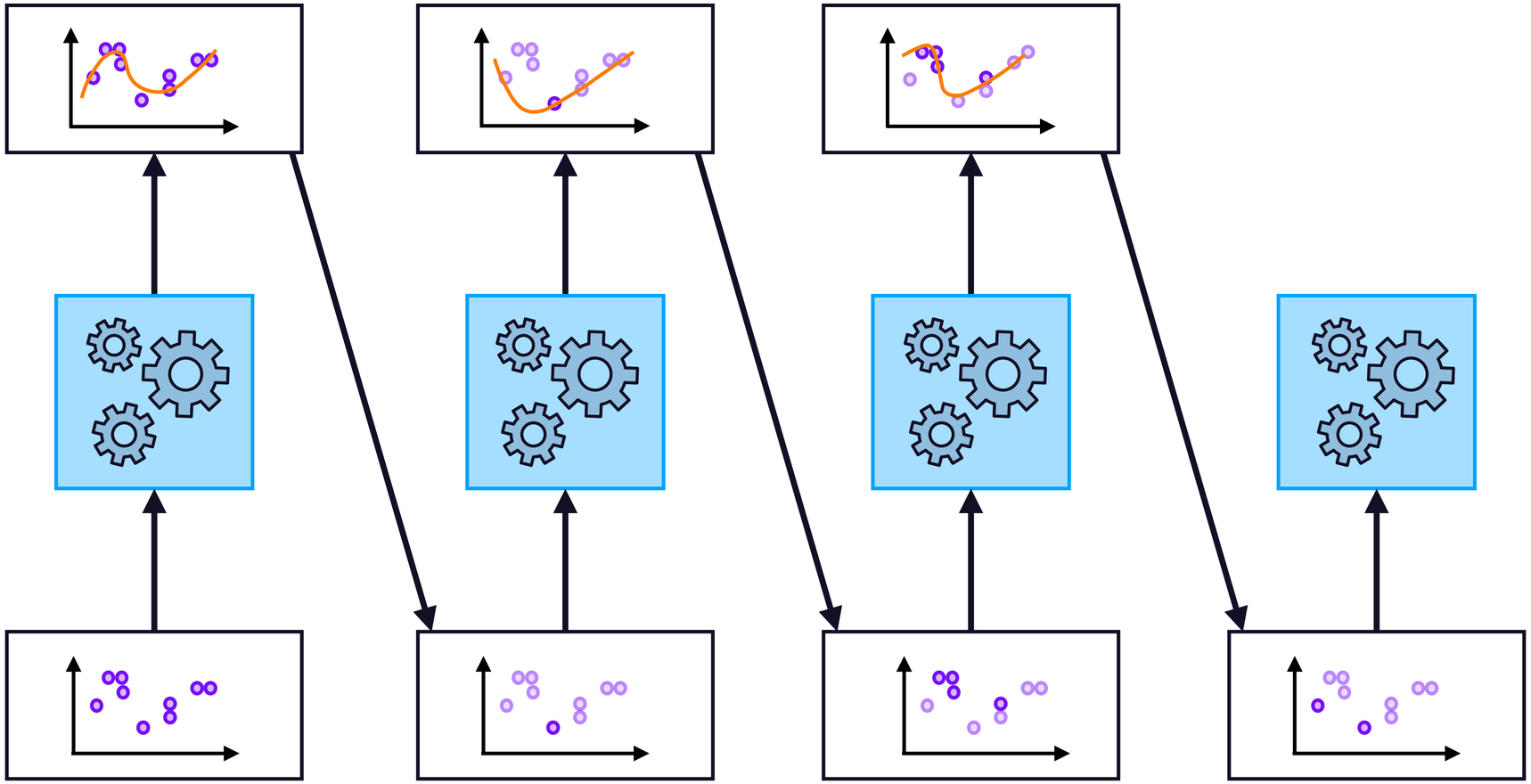
AdaBoost

Increase relative weights of those instances that previous predictors got wrong



AdaBoost

Limited room for parallelism as predictors require the results of previous predictors



AdaBoost

Final model output is weighted output of individual predictor outputs

More accurate predictors get higher weights

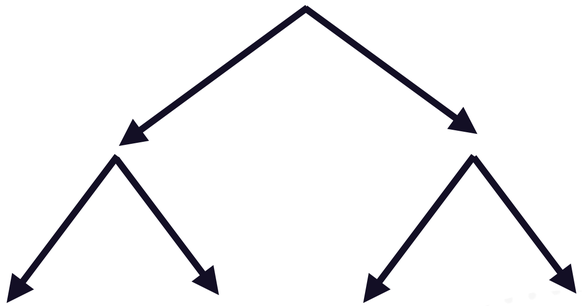
AdaBoost

Predictors that are wrong most often can have negative weights

Predictors that guess at random have weight close to zero

AdaBoost: First Learner

X1	X2	
A	0	Weight = 1/4
B	1	Weight = 1/4
C	1	Weight = 1/4
D	0	Weight = 1/4



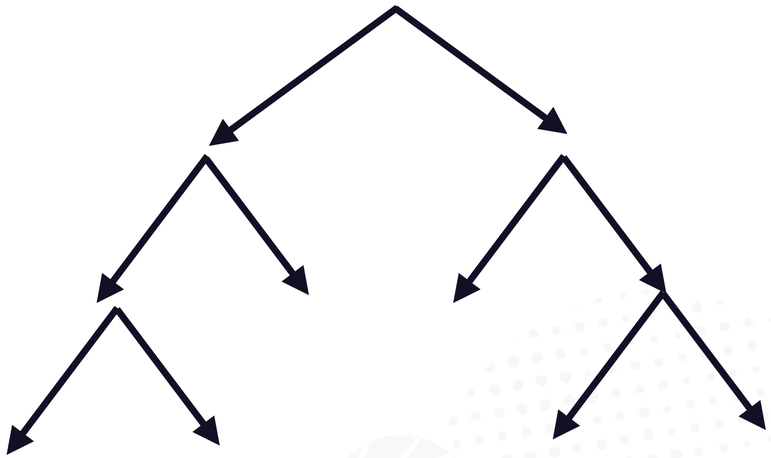
=

X1	X2	
A	0	Misclassified
B	1	Correctly classified
C	1	Misclassified
D	0	Correctly classified

AdaBoost: Second Learner

X1	X2
A	0
B	1
C	1
D	0

Weight = 3/8 (37.5%)
Weight = 1/8 (12.5%)
Weight = 3/8 (37.5%)
Weight = 1/8 (12.5%)

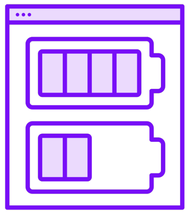


=

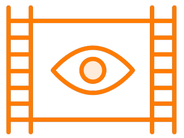
X	Y
A	0
B	1
C	1
D	0

Correctly classified
Correctly classified
Correctly classified
Misclassified

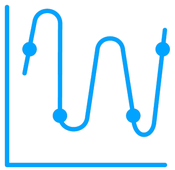
Advantages of AdaBoost



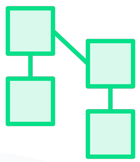
Boosts weak learners effectively



Automatically focuses on hard samples



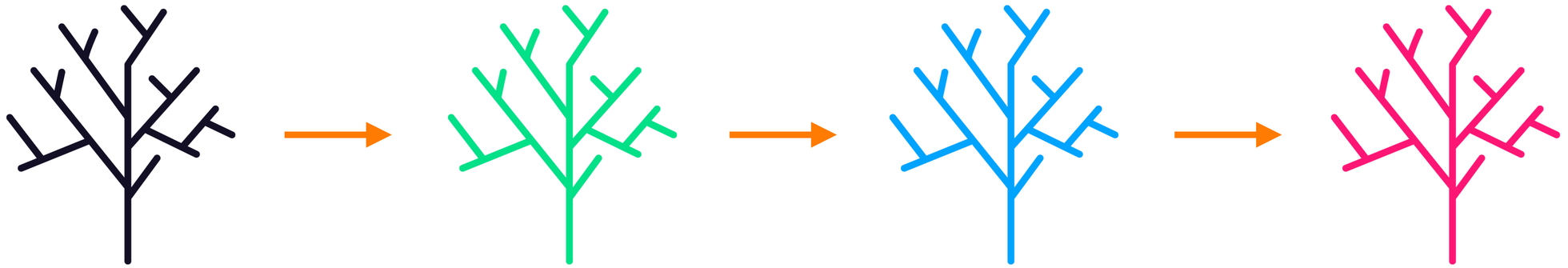
Surprisingly robust to overfitting



Simple and interpretable models

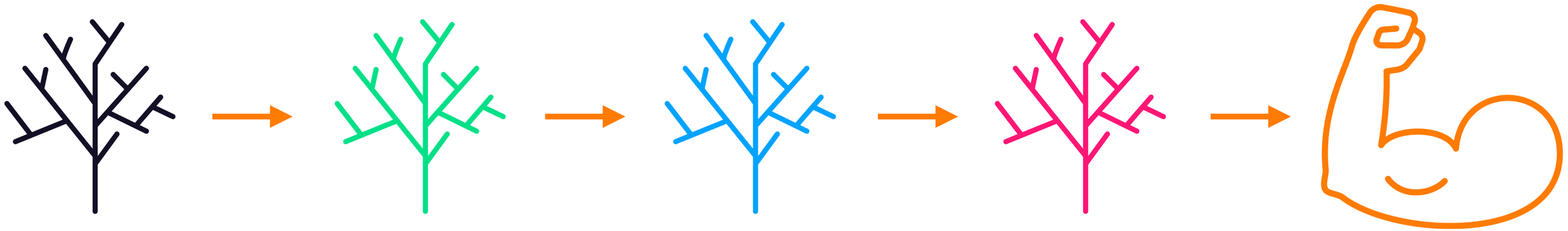
Gradient Boosting

Gradient Boosting



Many machine learning models come together
to work on the training data

Gradient Boosting



Many weak learners

Gradient Boosting with 3 Learners

Model 1:

$$y = F_1(x) + r_1$$

Model 2:

$$r_1 = F_2(x) + r_2$$

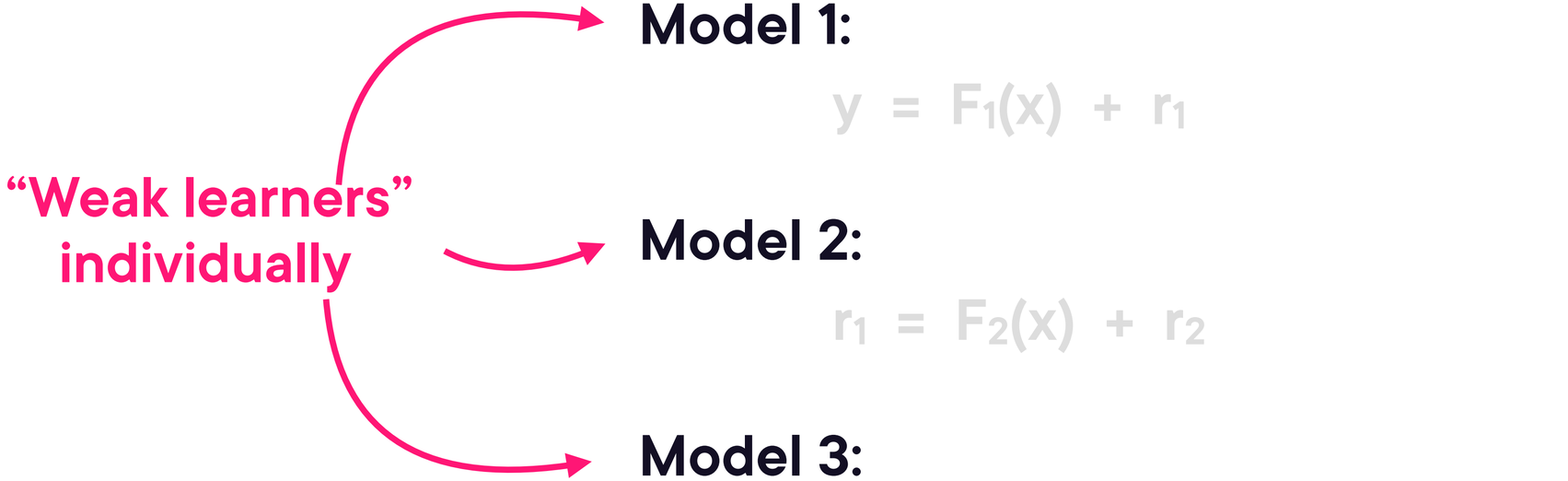
Model 3:

$$r_2 = F_3(x) + r_3$$

Combined Model:

$$Y = F_1(x) + \eta F_2(x) + \eta F_3(x) + r_3$$

Gradient Boosting with 3 Learners



Gradient Boosting

Model 1:

$$y = F_1(x) + r_1$$

Model 2:

$$r_1 = F_2(x) + r_2$$

Model 3:

$$r_2 = F_3(x) + r_3$$

Combined Model:

$$Y = F_1(x) + \eta F_2(x) + \eta F_3(x) + r_3$$

“Strong learner” when combined

Gradient Boosting

Model 1:

$$y = F_1(x) + r_1$$

Pseudo-residuals from Model 1

Model 2:

$$r_1 = F_2(x) + r_2$$

Model 3:

$$r_2 = F_3(x) + r_3$$

Combined Model:

$$Y = F_1(x) + \eta F_2(x) + \eta F_3(x) + r_3$$

Gradient Boosting

Model 1:

$$y = F_1(x) + r_1$$

Gradient of the loss function w.r.t. the model's prediction

Model 2:

$$r_1 = F_2(x) + r_2$$

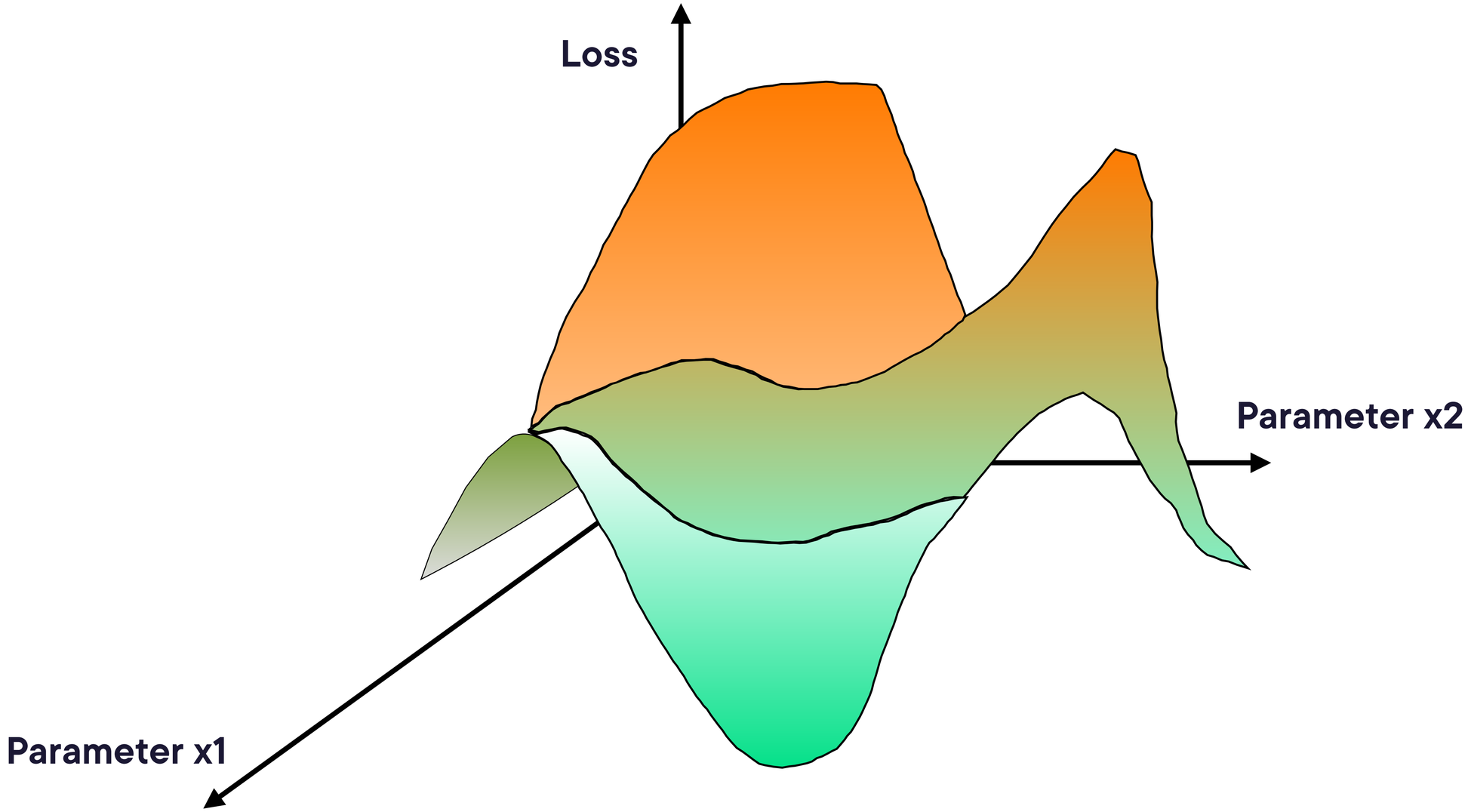
Model 3:

$$r_2 = F_3(x) + r_3$$

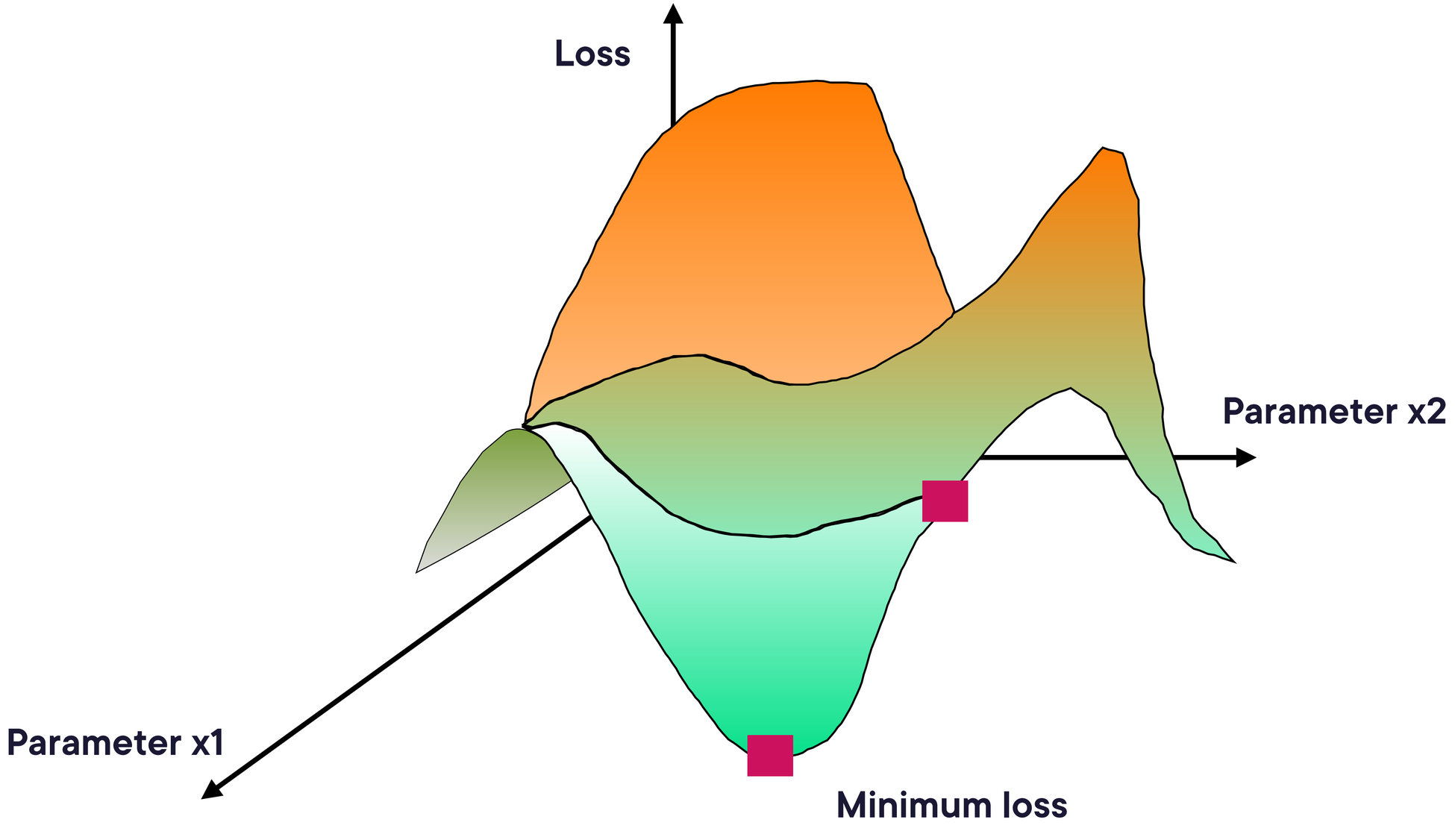
Combined Model:

$$Y = F_1(x) + \eta F_2(x) + \eta F_3(x) + r_3$$

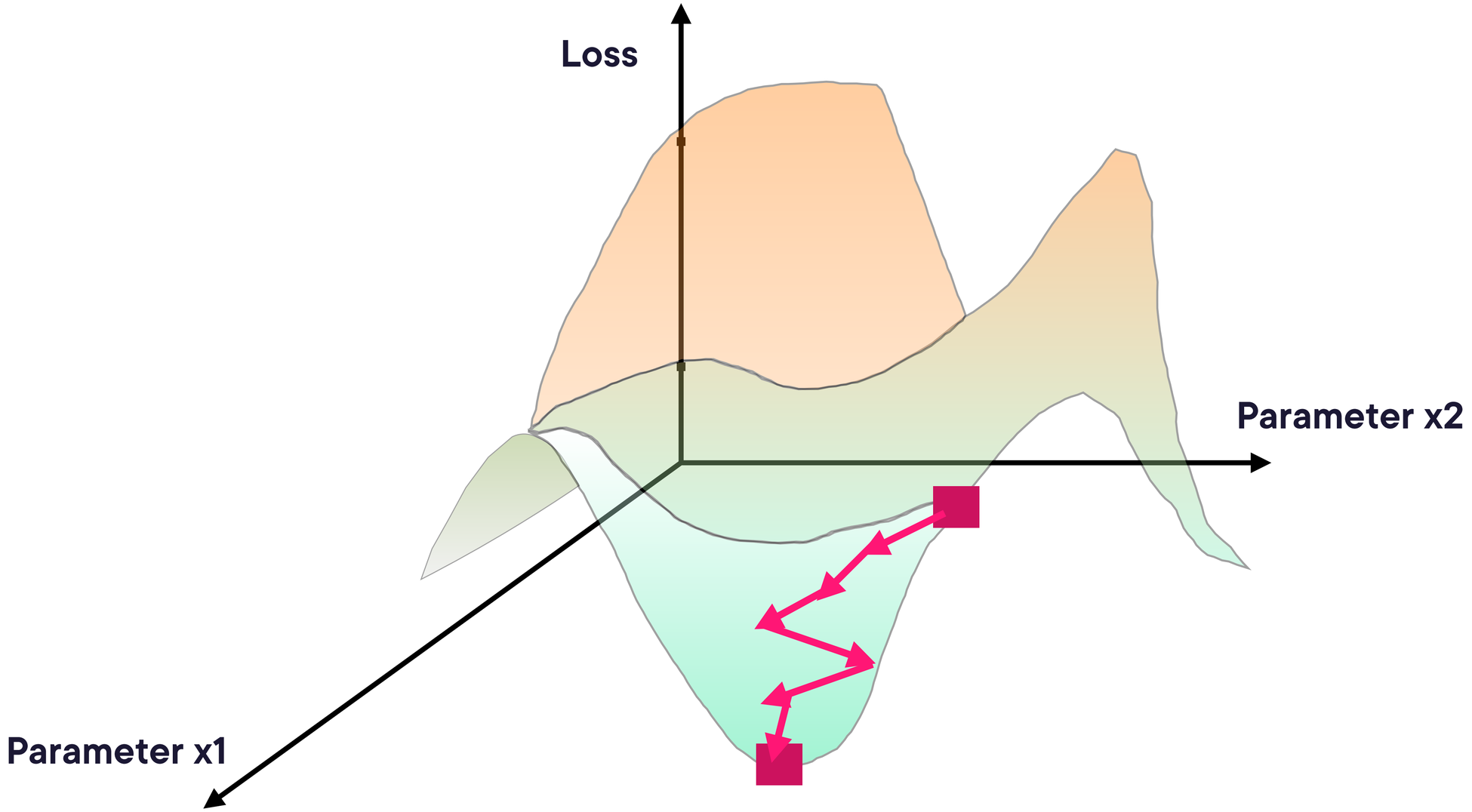
Loss Function as a Landscape



Loss Function as a Landscape



Take Steps to Minimize Loss



Gradient Boosting

Model 1:

$$y = F_1(x) + r_1$$

Pseudo-residuals from Model 1

What Model 1 fails to learn

Model 2:

$$r_1 = F_2(x) + r_2$$

Model 3:

$$r_2 = F_3(x) + r_3$$

Combined Model:

$$Y = F_1(x) + F_2(x) + F_3(x) + r_3$$

Gradient Boosting

Model 1:

$$y = F_1(x) + r_1$$

Pseudo-residuals from Model 1

Model 2:

$$r_1 = F_2(x) + r_2$$

What Model 1 fails to learn

Model 3:

$$r_2 = F_3(x) + r_3$$

Learn in Model 2

Combined Model:

$$Y = F_1(x) + \eta F_2(x) + \eta F_3(x) + r_3$$

Gradient Boosting

Model 1:

$$y = F_1(x) + r_1$$

Focuses on what previous model failed to learn

Model 2:

$$r_1 = F_2(x) + r_2$$

Model 3:

$$r_2 = F_3(x) + r_3$$

Combined Model:

$$Y = F_1(x) + \eta F_2(x) + \eta F_3(x) + r_3$$

Gradient Boosting

Model 1:

$$y = F_1(x) + r_1$$

Model 2:

$$r_1 = F_2(x) + r_2$$

Pseudo-residuals from Model 1 + 2

Model 3:

$$r_2 = F_3(x) + r_3$$

Combined Model:

$$Y = F_1(x) + \eta F_2(x) + \eta F_3(x) + r_3$$

Gradient Boosting

Model 1:

$$y = F_1(x) + r_1$$

Model 2:

$$r_1 = F_2(x) + r_2$$

Pseudo-residuals from Model 1 + 2

What Model 1 + 2 failed to learn

Model 3:

$$r_2 = F_3(x) + r_3$$

Combined Model:

$$Y = F_1(x) + \eta F_2(x) + \eta F_3(x) + r_3$$

Gradient Boosting

Model 1:

$$y = F_1(x) + r_1$$

Model 2:

$$r_1 = F_2(x) + r_2$$

Focuses on what previous models failed to learn

Model 3:

$$r_2 = F_3(x) + r_3$$

Combined Model:

$$Y = F_1(x) + \eta F_2(x) + \eta F_3(x) + r_3$$

Gradient Boosting

Model 1:

$$y = F_1(x) + r_1$$

Model 2:

$$r_1 = F_2(x) + r_2$$

Model 3:

$$r_2 = F_3(x) + r_3$$

Only these residuals are now unlearned

Combined Model:

$$Y = F_1(x) + \eta F_2(x) + \eta F_3(x) + r_3$$

Gradient Boosting

Model 1:

$$y = F_1(x) + r_1$$

Model 2:

$$r_1 = F_2(x) + r_2$$

Model 3:

$$r_2 = F_3(x) + r_3$$

Only these residuals are now unlearned

Combined Model:

$$Y = F_1(x) + \eta F_2(x) + \eta F_3(x) + r_3$$

Gradient Boosting

Model 1:

$$y = F_1(x) + r_1$$

Model 2:

$$r_1 = F_2(x) + r_2$$

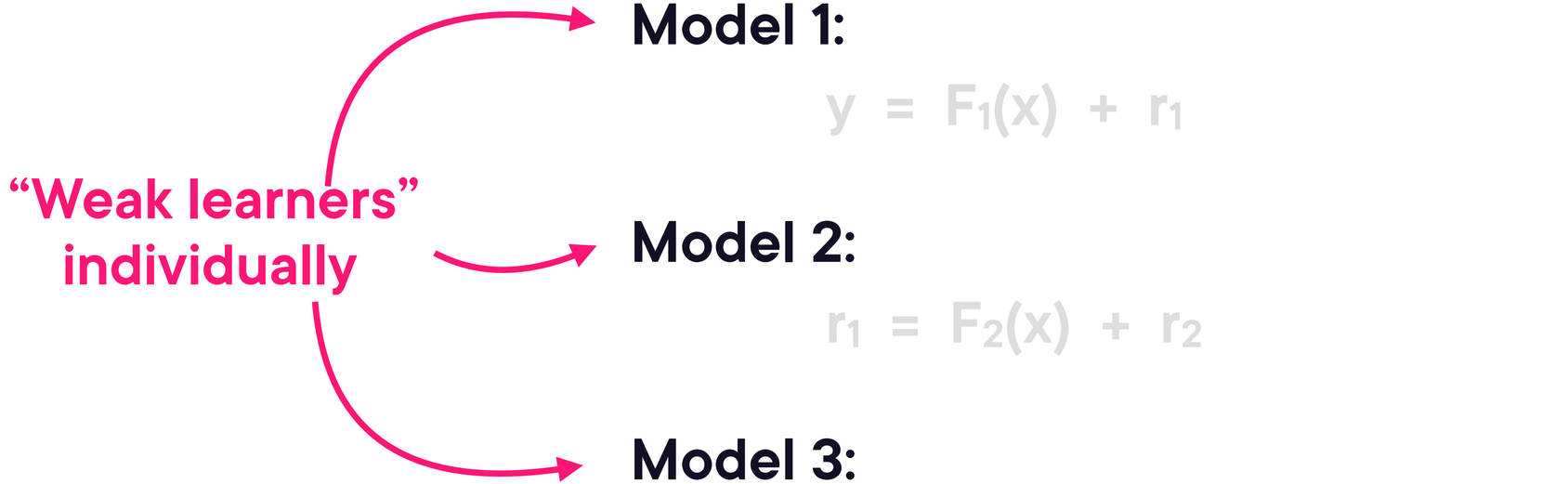
Model 3:

$$r_2 = F_3(x) + r_3$$

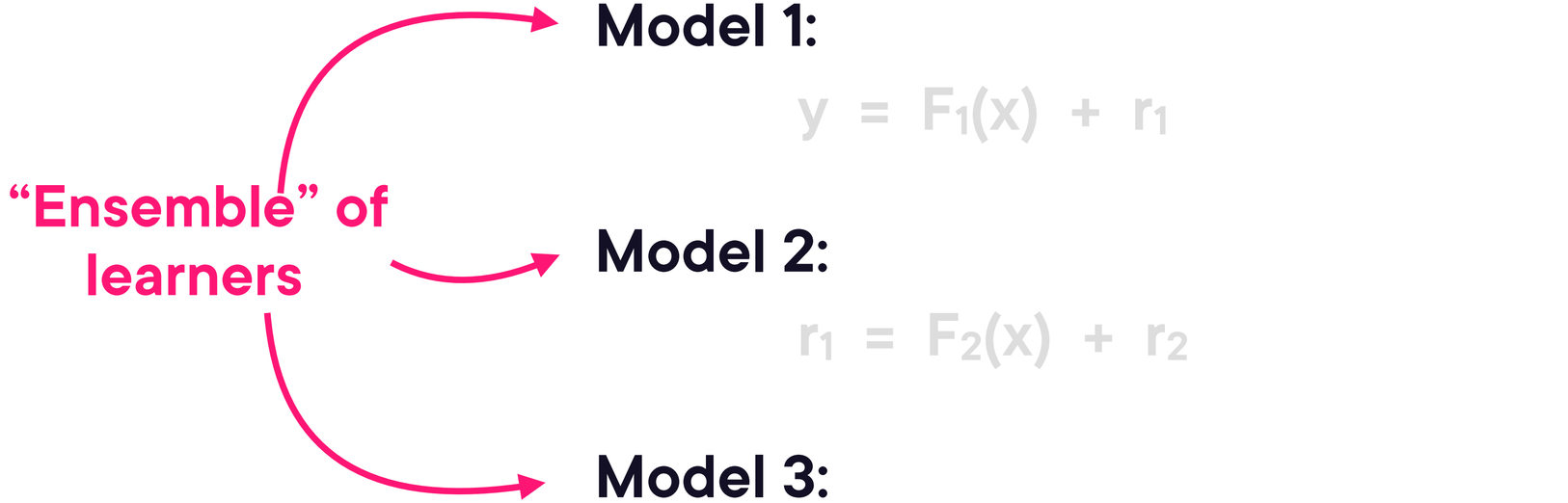
Combined Model:

$$Y = F_1(x) + \eta F_2(x) + \eta F_3(x) + r_3$$

Gradient Boosting



Gradient Boosting



Gradient Boosting

Model 1:

$$y = F_1(x) + r_1$$

Model 2:

$$r_1 = F_2(x) + r_2$$

Model 3:

$$r_2 = F_3(x) + r_3$$

Combined Model:

$$Y = F_1(x) + \eta F_2(x) + \eta F_3(x) + r_3$$

In practice:
100-200 weak learners, each learning from previous mistakes

Gradient Boosting

Model 1:

$$y = A_1 + B_1x + e_1$$

Model 2:

$$e_1 = A_2 + B_2x + e_2$$

Model 3:

$$e_2 = A_3 + B_3x + e_3$$

"Strong learner" when combined

Combined Model:

$$Y = F_1(x) + \eta F_2(x) + \eta F_3(x) + r_3$$

Gradient Boosting

Model 1:

$$y = A_1 + B_1x + e_1$$

Model 2:

$$e_1 = A_2 + B_2x + e_2$$

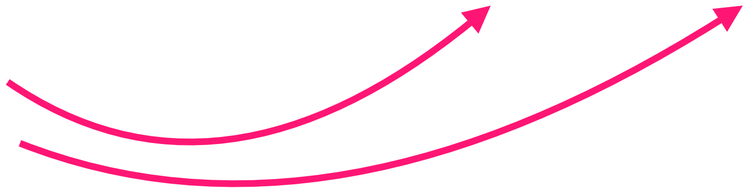
Model 3:

$$e_2 = A_3 + B_3x + e_3$$

Combined Model:

$$Y = F_1(x) + \eta F_2(x) + \eta F_3(x) + r_3$$

Each weak learner added to the existing model with a learning rate



Gradient Boosting

Model 1:

$$y = A_1 + B_1x + e_1$$

Model 2:

$$e_1 = A_2 + B_2x + e_2$$

Model 3:

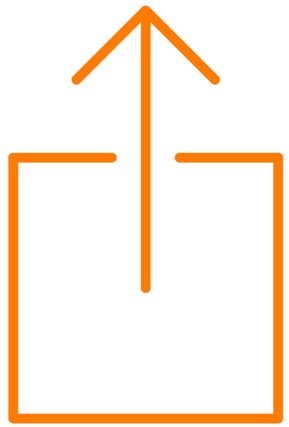
$$e_2 = A_3 + B_3x + e_3$$

Combined Model:

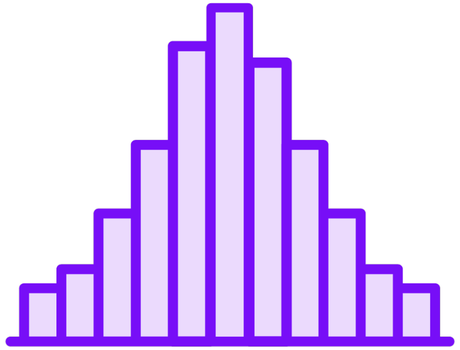
Y = Sum of outputs of weak learners

Variations of Gradient Boosting

Variations of Gradient Boosting



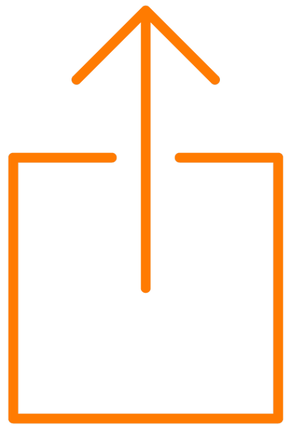
XGBoost



LightGBM

CatBoost

Variations of Gradient Boosting



XGBoost

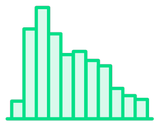
LightGBM

CatBoost

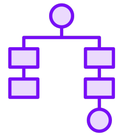
Innovations in XGBoost

Uses both the first derivative and the second derivative of the loss function

L1 and L2 regularization to control overfitting



Builds histogram of feature values to find tree splits

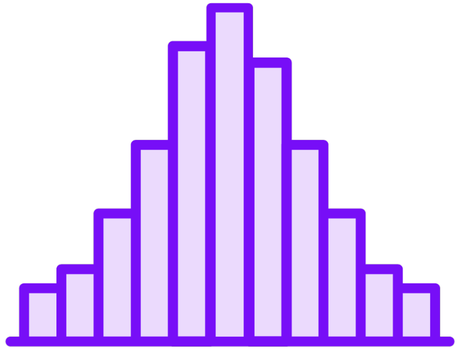


Parallelizes tree construction across features

Handles missing values automatically, no imputation needed

Variations of Gradient Boosting

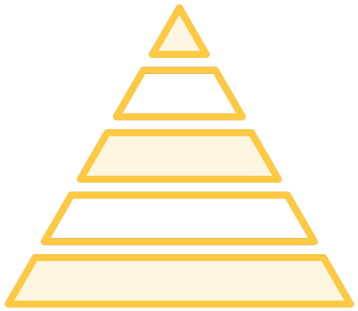
XGBoost



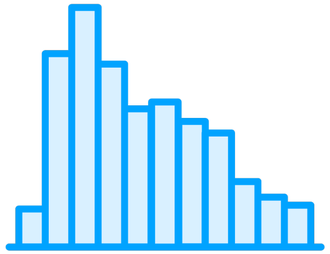
LightGBM

CatBoost

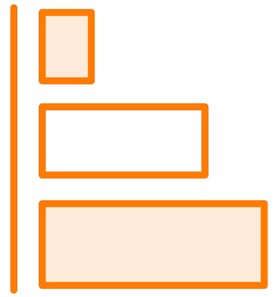
Innovations in LightGBM



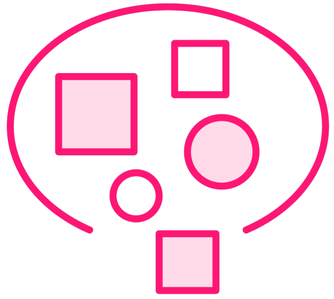
Leaf-wise tree growth



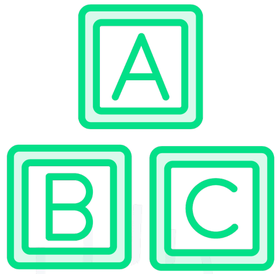
Histogram-based splitting



Gradient-based One-Side Sampling (GOSS)



Exclusive feature bundling



Categorical feature handling

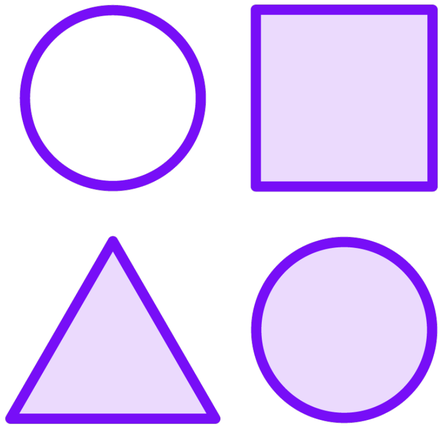
Variations of Gradient Boosting

XGBoost

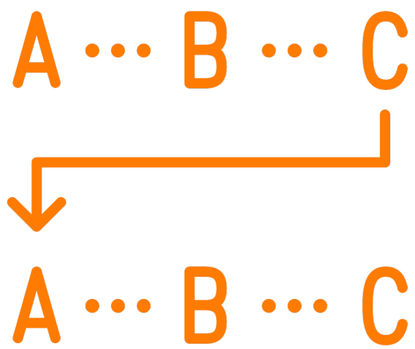
LightGBM

CatBoost

Innovations in CatBoost



Native handling of categorical features



Ordered boosting

Performs well out of the box

Symmetric (oblivious) trees

Ordered Boosting

When predicting for a data point, only use information that was available before that point in a random permutation of the dataset.