

# Boosting and Stacking

ADVANCED COMPUTING AND DIGITAL TECHNOLOGY RESEARCH CENTER

# Pattern Recognition and Machine Learning



## Lecture 14 Boosting and Stacking

**Chizhi Chris ZHANG**

zhangchizhi@ciomp.ac.cn

Advanced Computing and Digital Technology Research Center

University of Chinese Academy of Sciences

Autumn 2026

# Lecture plan and time allocation

ADVANCED COMPUTING AND DIGITAL TECHNOLOGY RESEARCH CENTER

| Topic                       | Minutes |
|-----------------------------|---------|
| AdaBoost                    | 45      |
| Gradient boosting           | 45      |
| Regularization and stacking | 45      |

## A new learner correcting the current model

Track sample weights; fit loss gradients; combine out-of-fold predictions.

The numerical steps are shown on the slides.



# The new learner adds a correction to the current score

ADVANCED COMPUTING AND DIGITAL TECHNOLOGY RESEARCH CENTER

The current classifier makes a specific set of mistakes.

An additive ensemble starts from a simple score  $F_0$  and adds one learner at a time. Earlier learners stay fixed while the next learner is fitted.

$$F_m(x) = F_{m-1}(x) + \alpha_m h_m(x), \quad F_M(x) = F_0(x) + \sum_{m=1}^M \alpha_m h_m(x).$$

$$F_0(x) = 2 \quad h_1(x) = 3$$

$$\alpha_1 = 0.2 \quad h_2(x) = -1$$

$$\alpha_2 = 0.5 \quad 2 + 0.6 - 0.5 = 2.1$$



# Five cases: start with equal observation weights

ADVANCED COMPUTING AND DIGITAL TECHNOLOGY RESEARCH CENTER

## Key idea

Use labels  $y_i \in \{-1, +1\}$  and predictions  $h_m(x_i) \in \{-1, +1\}$ . Observation weights  $w_{m,i}$  are nonnegative and sum to one.

$$y_i \in \{-1, +1\} \quad h_m(x_i) \in \{-1, +1\}$$

$$\epsilon_m = \sum_i w_{m,i} \mathbf{1}[y_i \neq h_m(x_i)].$$

$$w_{1,i} = 1/N \quad 0 < \epsilon_m < 0.5$$

The five labels and the first weak classifier are fixed below. We use this same example for three rounds, so every later weight and score can be checked.

| Case             | A   | B   | C   | D   | E   |
|------------------|-----|-----|-----|-----|-----|
| Label $y$        | +   | +   | -   | -   | +   |
| Classifier $h_1$ | +   | +   | -   | +   | +   |
| Initial weight   | 0.2 | 0.2 | 0.2 | 0.2 | 0.2 |

$$\epsilon_1 = 0.2$$

# Increase mistake weights and explicitly normalize

ADVANCED COMPUTING AND DIGITAL TECHNOLOGY RESEARCH CENTER

For the binary  $\pm 1$  convention, assign the weak classifier a coefficient based on its current weighted error.

$$\alpha_m = \frac{1}{2} \log \frac{1 - \epsilon_m}{\epsilon_m}, \quad \alpha_1 = \frac{1}{2} \log \frac{0.8}{0.2} = 0.6931.$$

$$F = \sum_m \alpha_m h_m$$

Update each weight according to whether its signed prediction is correct, then divide by the normalization constant  $Z_m$ .

## Worked calculation

$$w_{m+1,i} = \frac{w_{m,i} e^{-\alpha_m y_i h_m(x_i)}}{Z_m}, \quad Z_m = \sum_i w_{m,i} e^{-\alpha_m y_i h_m(x_i)}.$$

| Round 1 result                   | Correct A,B,C,E        | Wrong D               |
|----------------------------------|------------------------|-----------------------|
| Unnormalized weight              | $0.2e^{-0.6931} = 0.1$ | $0.2e^{0.6931} = 0.4$ |
| After normalization, $Z_1 = 0.8$ | 0.125 each             | 0.500                 |

## Round 2 moves weight from D to A

ADVANCED COMPUTING AND DIGITAL TECHNOLOGY RESEARCH CENTER

Round 2 starts from  $(.125, .125, .125, .5, .125)$ . Classifier  $h_2 = (-, +, -, -, +)$  fixes D but misses A.

**Error and coefficient**

$$\epsilon_2 = w_{2,A} = 1/8,$$

$$\alpha_2 = \frac{1}{2} \log \frac{7/8}{1/8} = \frac{1}{2} \log 7 \approx .9730.$$

Wrong cases multiply by  $\sqrt{7}$ ; correct cases by  $1/\sqrt{7}$ .

$$Z_2 = 2\sqrt{(1/8)(7/8)} = \sqrt{7}/4.$$

New weights sum to  $1/2 + 3/14 + 2/7 = 1$ . A now receives the greatest attention.

**Normalize all five weights**

$$w_{3,A} = \frac{(1/8)\sqrt{7}}{Z_2} = \frac{1}{2}, \quad w_{3,D} = \frac{(1/2)/\sqrt{7}}{Z_2} = \frac{2}{7}.$$

| Case       | Before round 2 | After round 2 |
|------------|----------------|---------------|
| A          | 1/8            | 1/2           |
| B,C,E each | 1/8            | 1/14          |
| D          | 1/2            | 2/7           |

## Round 3 uses the updated weights

ADVANCED COMPUTING AND DIGITAL TECHNOLOGY RESEARCH CENTER

Current weights are  $(1/2, 1/14, 1/14, 2/7, 1/14)$ . The third classifier predicts  $h_3 = (+, -, -, -, +)$ .

| Case       | A  | B   | C  | D  | E  |
|------------|----|-----|----|----|----|
| True label | +  | +   | -  | -  | +  |
| $h_3$      | +  | -   | -  | -  | +  |
| Incorrect? | No | Yes | No | No | No |

$$\epsilon_3 = w_{3,B} = 1/14, \quad \alpha_3 = \frac{1}{2} \log \frac{13/14}{1/14} = \frac{1}{2} \log 13 \approx 1.2825.$$

Only B is wrong. A, D and E were the different mistakes that received increased weight over these rounds.

$$F(x) = .6931h_1(x) + .9730h_2(x) + 1.2825h_3(x).$$

Next add these scores for each case before taking the final sign.



## Combine the coefficients before taking the sign

ADVANCED COMPUTING AND DIGITAL TECHNOLOGY RESEARCH CENTER

The following scores use the displayed four-decimal coefficients.

### What this tells us.

C has score  $-2.9486$  and E has  $+2.9486$ , also correct. Every member made one mistake, but this ensemble makes none on these five training cases. That is a training result, not a guaranteed test result.

$$F = 0.6931h_1 + 0.9730h_2 + 1.2825h_3 \quad F > 0$$

$$F < 0$$

| Case | Weighted score $F$                   | True label | Decision |
|------|--------------------------------------|------------|----------|
| A    | $0.6931 - 0.9730 + 1.2825 = 1.0026$  | +          | Correct  |
| B    | $0.6931 + 0.9730 - 1.2825 = 0.3836$  | +          | Correct  |
| D    | $0.6931 - 0.9730 - 1.2825 = -1.5624$ | -          | Correct  |

# Exponential loss pays attention to signed margins

ADVANCED COMPUTING AND DIGITAL TECHNOLOGY RESEARCH CENTER

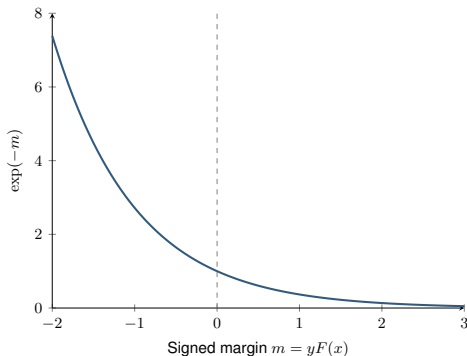
The signed margin  $m_i = y_i F(x_i)$  is positive for a correct decision and negative for a wrong decision. Exponential loss is  $L_i = e^{-m_i}$ .

$$m_i = y_i F(x_i) \quad L_i = e^{-m_i}$$

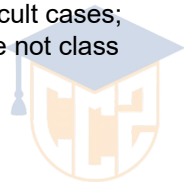
| Margin | Decision               | Exponential loss |
|--------|------------------------|------------------|
| -1     | Wrong                  | $e^1 = 2.718$    |
| 0      | On the boundary        | 1.000            |
| +1     | Correct                | $e^{-1} = 0.368$ |
| +3     | Correct, larger margin | $e^{-3} = 0.050$ |



# A negative margin receives a large exponential loss



The horizontal axis is the signed margin  $yF(x)$ . A negative margin means an incorrect sign and receives exponentially larger loss. This explains the reweighting of difficult cases; learner coefficients are not class probabilities.



# The coefficient minimizes the current exponential loss

ADVANCED COMPUTING AND DIGITAL TECHNOLOGY RESEARCH CENTER

Fix a learner with weighted error  $e$ . Correct cases carry mass  $1 - e$  and incorrect cases carry mass  $e$ . Their exponential-loss factor is a function of the learner coefficient  $\alpha$ .

$$e = .2$$

$$Z(\alpha) = (1 - e)e^{-\alpha} + e e^{\alpha}.$$

$$Z'(\alpha) = -(1 - e)e^{-\alpha} + e e^{\alpha} = 0 \Rightarrow e^{2\alpha} = \frac{e}{1 - e} \Rightarrow \alpha = \frac{1}{2} \log \frac{e}{1 - e} \approx .6931.$$

$$\alpha = \frac{1}{2} \log[(1 - e)/e]$$

With the learner fixed and normalized current weights, correct cases have total mass  $1 - e$ , and mistakes have mass  $e$ . The round objective becomes

$$\begin{aligned} Z^* &= (1 - e)\sqrt{\frac{e}{1 - e}} + e\sqrt{\frac{1 - e}{e}} \\ &= 2\sqrt{e(1 - e)}. \end{aligned}$$

$e = 0.2$ , so  $Z^* = 2\sqrt{0.2(0.8)} = 0.8$ . This equals the first round's four weights of  $.1$  plus one weight of  $.4$ .



# Weakness, noisy labels, and stopping belong together

ADVANCED COMPUTING AND DIGITAL TECHNOLOGY RESEARCH CENTER

A decision stump limits each correction to one split. Deeper weak trees can model interactions but may also chase difficult or mislabeled training cases more aggressively.

| Round count | Training error | Validation error |
|-------------|----------------|------------------|
| 10          | 0.12           | 0.16             |
| 50          | 0.02           | 0.11             |
| 200         | 0.00           | 0.14             |



# A regression correction predicts what is still missing

ADVANCED COMPUTING AND DIGITAL TECHNOLOGY RESEARCH CENTER

The current model already predicts a value or class score.

For squared-error regression,  $L(y, F) = \frac{1}{2}(y - F)^2$ . Its negative derivative with respect to the current prediction is the residual.

$$L(y, F) = \frac{1}{2}(y - F)^2$$

$$-\frac{\partial L}{\partial F} = y - F = r.$$



# The first tree fits two residual means

ADVANCED COMPUTING AND DIGITAL TECHNOLOGY RESEARCH CENTER

The constant ignores  $x$ . The first tree now looks for predictable structure in these residuals.

$$x = (1, 2, 3, 4) \quad y = (2, 4, 8, 10)$$

$$F_0 = \bar{y} = (2 + 4 + 8 + 10)/4 = 6.$$

| $x$             | 1  | 2  | 3 | 4  |
|-----------------|----|----|---|----|
| $y$             | 2  | 4  | 8 | 10 |
| $F_0$           | 6  | 6  | 6 | 6  |
| $r_1 = y - F_0$ | -4 | -2 | 2 | 4  |

$$\text{MSE}(F_0) = (16 + 4 + 4 + 16)/4 = 10.$$

Split at  $x = 2.5$ . A least-squares regression leaf predicts the mean residual among its training cases.

## Worked calculation

$$x = 2.5$$

$$h_1(x) = \begin{cases} (-4 - 2)/2 = -3, & x \leq 2.5, \\ (2 + 4)/2 = 3, & x > 2.5. \end{cases}$$

| $x$               | 1  | 2  | 3  | 4 |
|-------------------|----|----|----|---|
| $h_1$             | -3 | -3 | 3  | 3 |
| $F_1 = F_0 + h_1$ | 3  | 3  | 9  | 9 |
| New residual      | -1 | 1  | -1 | 1 |

# Shrinkage changes how much of the correction is used

ADVANCED COMPUTING AND DIGITAL TECHNOLOGY RESEARCH CENTER

This smaller step improves training less than the full step. Smaller learning rates usually need more rounds; validation decides whether the more conservative path helps.

$$\eta = .5 \quad F_1 = F_0 + .5h_1$$

| $x$       | 1    | 2    | 3   | 4   |
|-----------|------|------|-----|-----|
| $F_0$     | 6    | 6    | 6   | 6   |
| $.5h_1$   | -1.5 | -1.5 | 1.5 | 1.5 |
| $F_1$     | 4.5  | 4.5  | 7.5 | 7.5 |
| $y - F_1$ | -2.5 | -.5  | .5  | 2.5 |

$$\text{MSE} = (6.25 + .25 + .25 + 6.25)/4 = 3.25.$$



# The next tree must use the updated residuals

ADVANCED COMPUTING AND DIGITAL TECHNOLOGY RESEARCH CENTER

$\eta = 0.5$  throughout. Fit the new residuals, then add half of the second tree.

| $x$                            | 1    | 2    | 3    | 4    |
|--------------------------------|------|------|------|------|
| $y$                            | 2    | 4    | 8    | 10   |
| Current $F_1$                  | 4.5  | 4.5  | 7.5  | 7.5  |
| New residual $y - F_1$         | -2.5 | -0.5 | 0.5  | 2.5  |
| New Tree $h_2$ , threshold 2.5 | -1.5 | -1.5 | 1.5  | 1.5  |
| $F_2 = F_1 + 0.5h_2$           | 3.75 | 3.75 | 8.25 | 8.25 |

$$\text{MSE}(F_2) = \frac{(-1.75)^2 + 0.25^2 + (-0.25)^2 + 1.75^2}{4} = 1.5625.$$

$$x = 1 \quad 4.5 + 0.5(-1.5) = 3.75$$



# Other losses supply other pseudo-residuals

ADVANCED COMPUTING AND DIGITAL TECHNOLOGY RESEARCH CENTER

## Key idea

For a differentiable loss, compute one negative derivative per observation at the current ensemble prediction. Fit a regression learner to these pseudo-residuals.

$$r_{m,i} = - \left. \frac{\partial L(y_i, F)}{\partial F} \right|_{F=F_{m-1}(x_i)}, \quad F_m = F_{m-1} + \eta h_m.$$



# Calculate a classification correction direction

ADVANCED COMPUTING AND DIGITAL TECHNOLOGY RESEARCH CENTER

For labels  $y \in \{0, 1\}$ , let  $F(x)$  be a log-odds score and transform it to probability by  $p = \sigma(F)$ . Use binary cross-entropy.

$$y \in \{0, 1\} \quad p = \sigma(F)$$

$$p = \frac{1}{1 + e^{-F}},$$

$$L(y, F) = -y \log p - (1 - y) \log(1 - p).$$

$$\frac{\partial L}{\partial F} = p - y, \quad r = y - p.$$

Case A is positive with predicted probability 0.2; case B is negative with probability 0.8. Their pseudo-residuals push in opposite directions.

| Case | $y$ | $p$ | Negative Gradient<br>$y - p$ |
|------|-----|-----|------------------------------|
| A    | 1   | 0.2 | +0.8                         |
| B    | 0   | 0.8 | -0.8                         |

$$F_A = \log(0.2/0.8) = -1.386.$$

$$F_A^{\text{new}} = F_A + 0.1(0.8) = -1.306, \quad p_A^{\text{new}} \approx 0.213.$$

For B, similarly:  $F_B = \log 4 \approx 1.3863$  Update  
 $1.3863 + 0.1(-0.8) = 1.3063, p_B^{\text{new}} \approx 0.7869.$

# Multiclass gradients use one score for each class

ADVANCED COMPUTING AND DIGITAL TECHNOLOGY RESEARCH CENTER

For  $K$  classes, encode the label as a one-hot vector and maintain scores  $F_1, \dots, F_K$ . Softmax gives probabilities.

$$p_k = \frac{e^{F_k}}{\sum_{j=1}^K e^{F_j}}, \quad L = - \sum_k y_k \log p_k, \quad r_k = y_k - p_k.$$
$$y = (0, 1, 0), \quad p = (0.2, 0.5, 0.3), \quad r = (-0.2, +0.5, -0.3).$$



# Initialize classification from the training class rate

ADVANCED COMPUTING AND DIGITAL TECHNOLOGY RESEARCH CENTER

A constant binary model that minimizes cross-entropy predicts the training positive fraction  $\pi$ . In the log-odds convention, its starting score is  $F_0 = \log[\pi/(1 - \pi)]$ .

$$F_0 = \log[\pi/(1 - \pi)]$$

$$\pi = 0.10 \Rightarrow F_0 = \log(0.1/0.9) = -2.197, \quad \sigma(F_0) = 0.10.$$



# A compact implementation follows the squared-loss story

ADVANCED COMPUTING AND DIGITAL TECHNOLOGY RESEARCH CENTER

```
from sklearn.ensemble import GradientBoostingRegressor
gb = GradientBoostingRegressor(
    loss="squared_error", n_estimators=300,
    learning_rate=0.05, max_depth=2,
    subsample=0.8, validation_fraction=0.2,
    n_iter_no_change=20, random_state=14)
gb.fit(X_train, y_train)
y_pred = gb.predict(X_test)
```

**Read it as a process.** The model starts from a constant, fits successive correction trees, uses small updates, and stops according to an internal validation holdout.



# XGBoost adds curvature and explicit tree regularization

ADVANCED COMPUTING AND DIGITAL TECHNOLOGY RESEARCH CENTER

Gradient boosting chooses corrections from a loss gradient. XGBoost uses a local second-order loss approximation and adds a penalty on the new tree's leaves and scores.

$$\tilde{J} = \sum_i [g_i h(x_i) + \frac{1}{2} H_i h(x_i)^2] + \gamma T + \frac{\lambda}{2} \sum_{j=1}^T w_j^2.$$



# Compute both slope and curvature for a case

ADVANCED COMPUTING AND DIGITAL TECHNOLOGY RESEARCH CENTER

For target  $y = 1$  and current probability  $p = .2$ , differentiate cross-entropy with respect to score  $F$ . The gradient gives direction and the second derivative gives local curvature.

$$g = \frac{\partial L}{\partial F} = p - y = -0.8, \quad H = \frac{\partial^2 L}{\partial F^2} = p(1 - p) = 0.16.$$

If  $L = \frac{1}{2}(F - y)^2$ , take  $F = 6, y = 2$ , and

$$g = F - y = 4, \quad H = 1.$$

The second-order tree objective uses each observation's gradient  $g$  And Curvature  $H$ ; calculate these quantities for the selected loss.

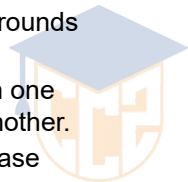


# Boosting controls act on different parts of the learning path

ADVANCED COMPUTING AND DIGITAL TECHNOLOGY RESEARCH CENTER

A correction can lower training loss while adding complexity. Choose controls together using the same validation task.

| Control                  | What changes?                       | Reason to compare                                             |
|--------------------------|-------------------------------------|---------------------------------------------------------------|
| Learning rate and rounds | Size and number of corrections      | Small steps with too few rounds can underfit.                 |
| Tree depth               | Interactions in each correction     | A stump cannot condition one feature's correction on another. |
| Leaf support             | Data behind each local estimate     | Very small leaves can chase noise.                            |
| Validation stopping      | Retained point on the learning path | Continuing to lower training loss may harm validation.        |



# Early stopping selects the learning path using validation

ADVANCED COMPUTING AND DIGITAL TECHNOLOGY RESEARCH CENTER

Compare validation loss while the boosting path develops.

| Round | Training loss | Validation loss |
|-------|---------------|-----------------|
| 20    | .42           | .46             |
| 60    | .27           | .34             |
| 100   | .18           | .32             |
| 160   | .10           | .36             |



# A leaf sums per-observation gradients and curvatures

ADVANCED COMPUTING AND DIGITAL TECHNOLOGY RESEARCH CENTER

## Key idea

All observations in leaf  $j$  receive the same score  $w_j$ . Define  $G_j = \sum_{i \in I_j} g_i$  and  $S_j = \sum_{i \in I_j} H_i$ . The leaf-dependent objective is quadratic.

$$G_j = \sum_{i \in I_j} g_i \quad S_j = \sum_{i \in I_j} H_i$$

$$J_j(w_j) = G_j w_j + \frac{1}{2} (S_j + \lambda) w_j^2 + \gamma.$$

$$\frac{dJ_j}{dw_j} = G_j + (S_j + \lambda) w_j = 0 \Rightarrow w_j^* = -\frac{G_j}{S_j + \lambda}.$$



The positive correction raises the class score for these positive cases.

Set  $\lambda = 1$ ,

$$Q(w) = -1.6w + \frac{1}{2} (0.32 + 1) w^2,$$

$$Q'(w) = -1.6 + 1.32w = 0 \Rightarrow w = 1.6/1.32 \approx 1.212.$$

This is a score correction; multiply by the learning rate before applying it.

## Regularization shrinks a leaf correction: use numbers

ADVANCED COMPUTING AND DIGITAL TECHNOLOGY RESEARCH CENTER

Two positive cases share a leaf, each with current probability 0.2. Thus  $G = -0.8 - 0.8 = -1.6$  and  $S = 0.16 + 0.16 = 0.32$ .

| Settings      | Leaf score $w^*$          | Applied correction with learning rate .1 |
|---------------|---------------------------|------------------------------------------|
| $\lambda = 0$ | $1.6/0.32 = 5$            | 0.5                                      |
| $\lambda = 1$ | $1.6/1.32 \approx 1.2121$ | 0.12121                                  |



### Worked calculation

Original probability .2 corresponds to  $F = \log(0.2/0.8) \approx -1.3863$ . After the second row's update,  $F \approx -1.2651$ , probability  $\sigma(F) \approx 0.2201$ .

# A split must earn its extra leaf

ADVANCED COMPUTING AND DIGITAL TECHNOLOGY RESEARCH CENTER

Compare splitting one leaf into left and right leaves. The improvement in the optimal second-order objective is

$$\text{Gain} = \frac{1}{2} \left[ \frac{G_L^2}{S_L + \lambda} + \frac{G_R^2}{S_R + \lambda} - \frac{(G_L + G_R)^2}{S_L + S_R + \lambda} \right] - \gamma.$$

$$G_L = -4 \quad S_L = 2$$

$$G_R = +4 \quad S_R = 2$$

$$\lambda = 1 \quad \gamma = 1$$

$$\frac{1}{2}(16/3 + 16/3) - 1 = 4.333$$

Here the parent gradient is  $G_L + G_R = 0$ , so its score term is zero. The two children can correct opposite residual directions. Subtract  $\gamma$  once because one leaf becomes two.

With the same gradients but  $\gamma = 6$ ,  $\text{Gain} = 5.333 - 6 = -.667$ : reject this split.



# Histogram bins reduce the split-search workload

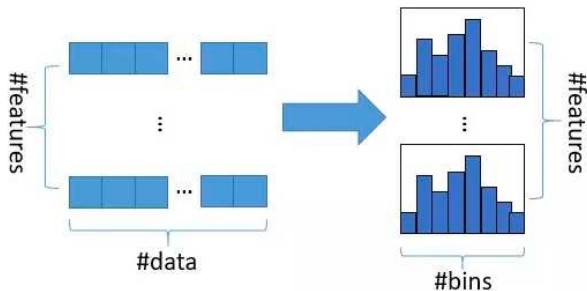
ADVANCED COMPUTING AND DIGITAL TECHNOLOGY RESEARCH CENTER

Exact split search sorts distinct feature values and checks many thresholds. Histogram methods map values into a limited number of bins and accumulate gradient and curvature sums per bin.

|                   |                  |                  |                  |
|-------------------|------------------|------------------|------------------|
| Feature values    | 1.0, 1.1, 1.2    | 2.8, 2.9         | 5.0, 5.1         |
| Interval          | 1                | 2                | 3                |
| Stored statistics | $\sum g, \sum H$ | $\sum g, \sum H$ | $\sum g, \sum H$ |

# Histogram bins reduce candidate split positions

ADVANCED COMPUTING AND DIGITAL TECHNOLOGY RESEARCH CENTER



The left column contains individual feature values; the right groups them into bins. Split search aggregates gradients and curvatures by bin, reducing candidate positions at the cost of coarser thresholds.



## Choose the leaf with the largest available gain

ADVANCED COMPUTING AND DIGITAL TECHNOLOGY RESEARCH CENTER

Level-wise growth splits nodes broadly at one depth. Leaf-wise growth chooses the current leaf with the largest available gain, so the tree can become asymmetric.

| Current candidate leaf | Best candidate split gain |
|------------------------|---------------------------|
| Left leaf.             | 0.4                       |
| Right leaf.            | 0.8                       |
| Other leaves           | 0.2                       |

The right leaf has gain .8, so split it next. Recompute candidate gains after the split.

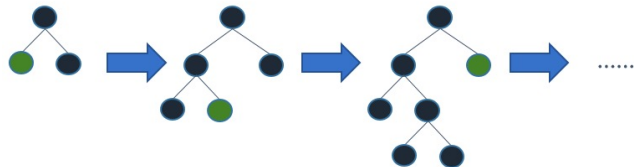
**Check.** Limit leaf count, depth and minimum samples per leaf to avoid repeatedly fitting chance patterns in small groups.



# Leaf-wise growth expands the most useful current leaf

ADVANCED COMPUTING AND DIGITAL TECHNOLOGY RESEARCH CENTER

At each step, compare the available gains across current leaves and split the best one. The highlighted branch can grow deeper than its neighbors. Depth and minimum-leaf constraints control this concentration of capacity.



Leaf-wise tree growth



# Categorical target statistics can leak the answer

ADVANCED COMPUTING AND DIGITAL TECHNOLOGY RESEARCH CENTER

A category mean can be a useful numerical feature, but a row's own label must not determine its training encoding. For a rare category occurring once, an unprotected category mean equals its label.

$$\text{Smoothed mean } (c) = \frac{\sum_{i \in c} y_i + a\pi}{n_c + a}.$$



## Own-target leakage

A training row must not use its own target to construct its category encoding.

# Stacking trains on out-of-fold base predictions

ADVANCED COMPUTING AND DIGITAL TECHNOLOGY RESEARCH CENTER

Each row's prediction must come from a fit that excluded that row.

| Fold | Base-model training rows | Rows predicted |
|------|--------------------------|----------------|
| 1    | C, D, E, F               | A, B           |
| 2    | A, B, E, F               | C, D           |
| 3    | A, B, C, D               | E, F           |

Restore predictions to the original A–F row order. Each base learner supplies a column. Train the meta-model on these columns and verified labels, excluding the final test set.



# In-sample stacking teaches the meta-model a false lesson

ADVANCED COMPUTING AND DIGITAL TECHNOLOGY RESEARCH CENTER

Three held-out folds are A/B, C/D and E/F. The two base fits for a row exclude that row.

| Rows | Fold | Tree Model $p$ | Logistic regression $p$ | $y$ |
|------|------|----------------|-------------------------|-----|
| A    | 1    | .70            | .60                     | 1   |
| B    | 1    | .30            | .40                     | 0   |
| C    | 2    | .80            | .65                     | 1   |
| D    | 2    | .45            | .20                     | 0   |
| E    | 3    | .55            | .75                     | 1   |
| F    | 3    | .25            | .35                     | 0   |

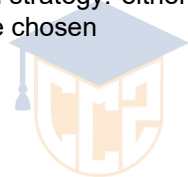


A deep tree can nearly memorize the training labels. Compare its apparent training probabilities with genuinely held-out probabilities.

# Prediction must recreate the meta-model inputs

ADVANCED COMPUTING AND DIGITAL TECHNOLOGY RESEARCH CENTER

After fitting the meta-learner, produce new-case base predictions using a defined strategy: either average the fold models or refit each base learner on all training data. Follow the chosen implementation consistently.



$$\mathbf{z}_{\text{new}} = (\mathbf{p}_{\text{tree}}, \mathbf{p}_{\text{logistic}}), \quad \mathbf{p}_{\text{final}} = \sigma(\beta_0 + \beta_1 \mathbf{p}_{\text{tree}} + \beta_2 \mathbf{p}_{\text{logistic}}).$$
$$\mathbf{z} = (0.70, 0.60) \quad -2 + 1.4 + 0.6 = 0$$

# Reconstruct the correction, then defend the combination

ADVANCED COMPUTING AND DIGITAL TECHNOLOGY RESEARCH CENTER

For each method, identify which current error supplies the next training signal. In stacking, the crucial evidence is how base predictions were obtained: every meta-training row needs predictions from a fit that excluded it.

| Question                                 | What a complete answer includes                                                            |
|------------------------------------------|--------------------------------------------------------------------------------------------|
| AdaBoost: one wrong weight 0.2           | Weighted error .2; $\alpha = 0.6931$ ; after normalization the mistaken case has weight .5 |
| Regression: $y = 10$ Current $F = 6$     | Squared-loss residual +4 , then add the correction                                         |
| Binary: $y = 1, p = 0.2$                 | Negative Gradient +0.8 , update the score and apply sigmoid                                |
| Stacking: deep-tree training predictions | Replace them with OOF predictions before meta-training                                     |