Lecture 23

More on XGBoost

COURSE MATH 637: Mathematical Techniques in Data Science

WEBSITE https://jacobcalvert.com/teaching/math637fall2026/

INSTRUCTOR Jacob Calvert (calvert@udel.edu)

OFFICE HOURS MW 2:45pm–3:45pm, Ewing Hall 509

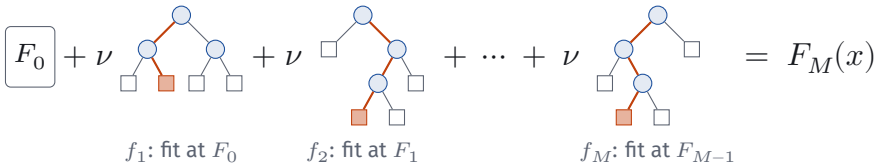
Announcements & Reminders

- HW5 is a 1-page final project proposal, due by class next Wed. (Nov. 11th)
- For full credit, describe project idea in enough detail that I can reasonably evaluate its suitability. In particular, address:
  1. Will you work alone, or with a small team? Name team members, explain division of work. More will be expected of larger groups.
  2. What question(s) will you try to answer?
  3. What dataset(s) will you use? Provide specifications, e.g., # examples, features and their types, data missingness, etc.
  4. What simple baseline technique(s) will you use, e.g., for data exploration and prediction/inference?
  5. What ideas from the course/literature will you use beyond these baselines? (Note: we'll cover neural net–based methods soon.)
  6. What is the ideal outcome? The main piftall?

Plan for today

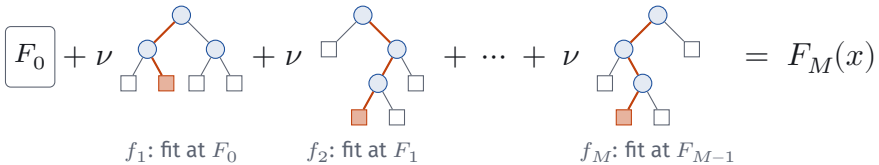
• Recap: one round, and what XGBoost changes	10 min
• One leaf: its best correction	6 min
• One leaf or two? The gain	14 min
• Histograms: searching thresholds fast	12 min
• LightGBM	14 min
• In practice, and two questions	12 min

Recall: Boosting adds trees



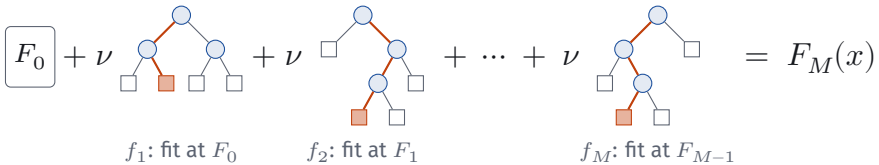
- each tree is a small correction to the current score F, fit to the derivatives of the loss at that score

Recall: Boosting adds trees



- each tree is a small correction to the current score F, fit to the derivatives of the loss at that score
- to obtain a score for x, score x using each tree separately, then add up the values times ν

Recall: Boosting adds trees



- each tree is a small correction to the current score F, fit to the derivatives of the loss at that score
- to obtain a score for x, score x using each tree separately, then add up the values times ν
- for classification with log loss, the score F is the log-odds, and $p(x) = \sigma(F_M(x))$

Recall: Gradient boosting with log loss

Definition (Gradient boosting with log loss (L21–22))

Start from $F_0 = \log(\bar{y}/(1-\bar{y}))$. For $m = 1, \dots, M$:

1. compute $p_i = \sigma(F_{m-1}(x_i))$ and the targets $r_i = y_i - p_i$
2. fit a least-squares tree to the (x_i, r_i), keep the leaves R_{jm}
3. add the tree $f(x) = \sum_j w_j \mathbf{1}\{x \in R_{jm}\}$ with leaf values

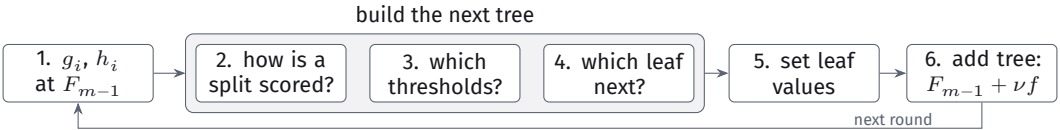
$$w_j = \sum_{R_{jm}} (y_i - p_i) \Big/ \sum_{R_{jm}} p_i(1-p_i),$$

as $F_m(x) = F_{m-1}(x) + \nu f(x)$.

Estimate $p_M(x) = \sigma(F_M(x))$.

Today's notation: $r_i = -g_i$ and $w_j = -G_j/H_j$.

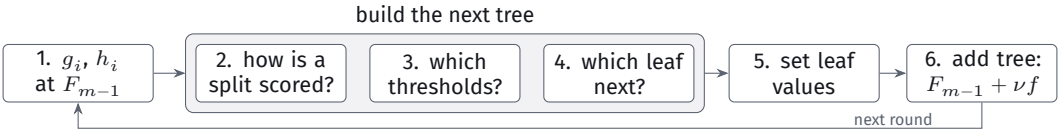
Recall: One round of gradient boosting



The answers to 2–5 differentiate regular gradient boosting from XGBoost and LightGBM. For regular gradient boosting the answers are:

2. the decrease in the RSS of the targets $-g_i$

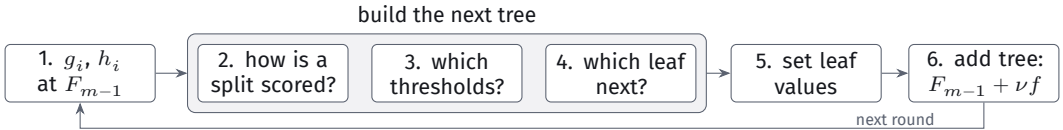
Recall: One round of gradient boosting



The answers to 2–5 differentiate regular gradient boosting from XGBoost and LightGBM. For regular gradient boosting the answers are:

2. the decrease in the RSS of the targets $-g_i$
3. every distinct value of each feature

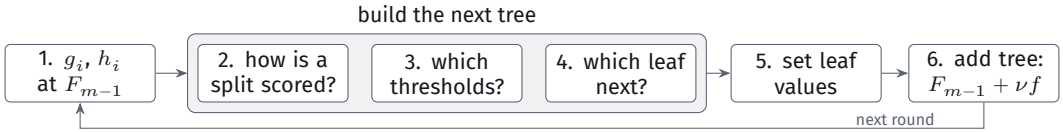
Recall: One round of gradient boosting



The answers to 2–5 differentiate regular gradient boosting from XGBoost and LightGBM. For regular gradient boosting the answers are:

2. the decrease in the RSS of the targets $-g_i$
3. every distinct value of each feature
4. every leaf at one depth, down to a fixed depth

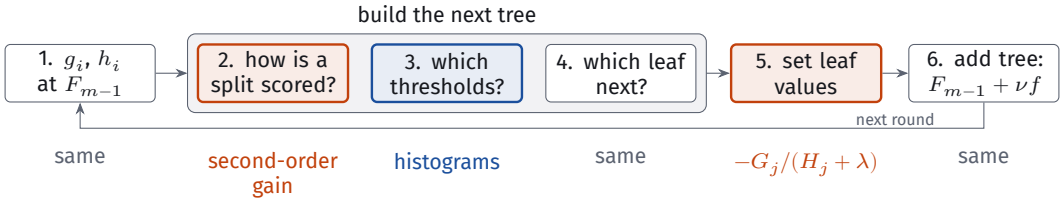
Recall: One round of gradient boosting



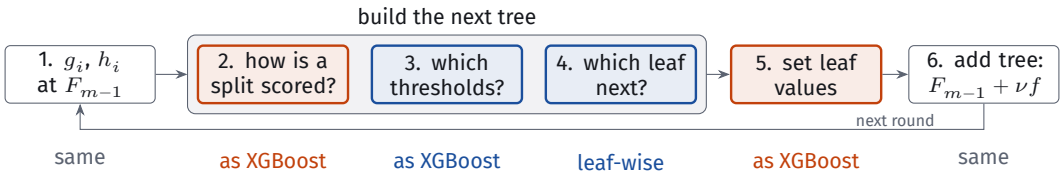
The answers to 2–5 differentiate regular gradient boosting from XGBoost and LightGBM. For regular gradient boosting the answers are:

2. the decrease in the RSS of the targets $-g_i$
3. every distinct value of each feature
4. every leaf at one depth, down to a fixed depth
5. $w_j = -G_j/H_j$

What XGBoost changes



Ahead: What LightGBM changes



Recall: XGBoost's objective

Round m chooses the tree f, with T leaves R_j and leaf values w_j, to minimize

$$\sum_i L(y_i, F_{m-1}(x_i) + f(x_i)) + \gamma T + \tfrac{1}{2}\lambda \sum_{j=1}^{T} w_j^2.$$

Expand each loss to second order around $F_{m-1}(x_i)$, with $g_i = \frac{\partial L}{\partial F}$ and $h_i = \frac{\partial^2 L}{\partial F^2}$ (log loss: $g_i = p_i - y_i$, $h_i = p_i(1-p_i)$), and sum over each leaf:

$$\text{const} + \sum_{j=1}^{T} \left[G_j\, w_j + \tfrac{1}{2}(H_j + \lambda) {w_j}^2 \right] + \gamma T, \qquad G_j = \sum_{x_i \in R_j} g_i, \quad H_j = \sum_{x_i \in R_j} h_i.$$

Recall: XGBoost's objective

Round m chooses the tree f, with T leaves R_j and leaf values w_j, to minimize

$$\sum_i L(y_i, F_{m-1}(x_i) + f(x_i)) + \gamma T + \tfrac{1}{2}\lambda \sum_{j=1}^{T} w_j^2.$$

Expand each loss to second order around $F_{m-1}(x_i)$, with $g_i = \frac{\partial L}{\partial F}$ and $h_i = \frac{\partial^2 L}{\partial F^2}$ (log loss: $g_i = p_i - y_i$, $h_i = p_i(1-p_i)$), and sum over each leaf:

$$\text{const} + \sum_{j=1}^{T} \left[G_j\, w_j + \tfrac{1}{2}(H_j + \lambda) {w_j}^2 \right] + \gamma T, \qquad G_j = \sum_{x_i \in R_j} g_i, \quad H_j = \sum_{x_i \in R_j} h_i.$$

Key idea: One objective now decides both the splits and the leaf values.

Plan for today

• Recap: one round, and what XGBoost changes	10 min
• One leaf: its best correction	6 min
• One leaf or two? The gain	14 min
• Histograms: searching thresholds fast	12 min
• LightGBM	14 min
• In practice, and two questions	12 min

XGBoost's leaf values



box 5: set the leaf values

Proposition

For fixed leaves $R_1, \dots, R_T$ with every $H_j + \lambda > 0$, the approximate objective is smallest at

$$w_j^* = -\frac{G_j}{H_j + \lambda} \qquad \text{with value} \qquad \text{const} - \sum_{j=1}^{T} \frac{G_j^2}{2(H_j + \lambda)} + \gamma T.$$

Example: XGBoost's leaf values for four points

Recall: Start from constant $F_0 = \log(\bar{y}/(1-\bar{y}))$, probabilities are $p_i = \sigma(F_{m-1}(x_i))$, gradients and Hessians are $g_i = p_i - y_i$ and $h_i = p_i(1-p_i)$, their sums over leaf j are G_j and H_j, set leaf values to $w_j = -G_j/(H_j+\lambda)$.

Example

$x = (-2, -1, 1, 2)$, $y = (0, 0, 1, 1)$, log loss, one split at $x = 0$. In round 1, find each g_i and h_i, and the leaf values for $\lambda = 1$.

Example: XGBoost's leaf values for four points

Recall: Start from constant $F_0 = \log(\bar{y}/(1-\bar{y}))$, probabilities are $p_i = \sigma(F_{m-1}(x_i))$, gradients and Hessians are $g_i = p_i - y_i$ and $h_i = p_i(1-p_i)$, their sums over leaf j are G_j and H_j, set leaf values to $w_j = -G_j/(H_j+\lambda)$.

Example

$x = (-2, -1, 1, 2)$, $y = (0, 0, 1, 1)$, log loss, one split at $x = 0$. In round 1, find each g_i and h_i, and the leaf values for $\lambda = 1$.

- $\bar{y} = \frac{1}{2}$, so $F_0 = 0$ and every $p_i = \frac{1}{2}$.

Example: XGBoost's leaf values for four points

Recall: Start from constant $F_0 = \log(\bar{y}/(1-\bar{y}))$, probabilities are $p_i = \sigma(F_{m-1}(x_i))$, gradients and Hessians are $g_i = p_i - y_i$ and $h_i = p_i(1-p_i)$, their sums over leaf j are G_j and H_j, set leaf values to $w_j = -G_j/(H_j+\lambda)$.

Example

$x = (-2, -1, 1, 2)$, $y = (0, 0, 1, 1)$, log loss, one split at $x = 0$. In round 1, find each g_i and h_i, and the leaf values for $\lambda = 1$.

- $\bar{y} = \frac{1}{2}$, so $F_0 = 0$ and every $p_i = \frac{1}{2}$.
- This gives $g = (\frac{1}{2}, \frac{1}{2}, -\frac{1}{2}, -\frac{1}{2})$ and every $h_i = \frac{1}{4}$.

Example: XGBoost's leaf values for four points

Recall: Start from constant $F_0 = \log(\bar{y}/(1-\bar{y}))$, probabilities are $p_i = \sigma(F_{m-1}(x_i))$, gradients and Hessians are $g_i = p_i - y_i$ and $h_i = p_i(1-p_i)$, their sums over leaf j are G_j and H_j, set leaf values to $w_j = -G_j/(H_j+\lambda)$.

Example

$x = (-2, -1, 1, 2)$, $y = (0, 0, 1, 1)$, log loss, one split at $x = 0$. In round 1, find each g_i and h_i, and the leaf values for $\lambda = 1$.

- $\bar{y} = \frac{1}{2}$, so $F_0 = 0$ and every $p_i = \frac{1}{2}$.
- This gives $g = (\frac{1}{2}, \frac{1}{2}, -\frac{1}{2}, -\frac{1}{2})$ and every $h_i = \frac{1}{4}$.
- Left of split: $G_L = 1$, $H_L = \frac{1}{2}$. Right of split: $G_R = -1$, $H_R = \frac{1}{2}$.

Example: XGBoost's leaf values for four points

Recall: Start from constant $F_0 = \log(\bar{y}/(1-\bar{y}))$, probabilities are $p_i = \sigma(F_{m-1}(x_i))$, gradients and Hessians are $g_i = p_i - y_i$ and $h_i = p_i(1-p_i)$, their sums over leaf j are G_j and H_j, set leaf values to $w_j = -G_j/(H_j+\lambda)$.

Example

$x = (-2, -1, 1, 2)$, $y = (0, 0, 1, 1)$, log loss, one split at $x = 0$. In round 1, find each g_i and h_i, and the leaf values for $\lambda = 1$.

- $\bar{y} = \frac{1}{2}$, so $F_0 = 0$ and every $p_i = \frac{1}{2}$.
- This gives $g = (\frac{1}{2}, \frac{1}{2}, -\frac{1}{2}, -\frac{1}{2})$ and every $h_i = \frac{1}{4}$.
- Left of split: $G_L = 1$, $H_L = \frac{1}{2}$. Right of split: $G_R = -1$, $H_R = \frac{1}{2}$.
- When $\lambda = 1$, $w_L = -2/3$ and $w_R = 2/3$.

Example: What λ does to the leaves

Credit data using only balance. The four leaves of round 1. Each account has $h_i = p(1-p) = 0.032$. Leaf values $w = -G/(H+\lambda)$:

leaf	H	$\lambda = 0$	$\lambda = 1$	$\lambda = 10$
$4{,}176$ accounts (15 defaults)	134.0	-0.92	-0.92	-0.86
604 accounts (54 defaults)	19.4	1.75	1.67	1.16
160 accounts (55 defaults)	5.1	9.68	8.10	3.28
60 accounts (42 defaults)	1.9	20.77	13.67	3.35

- $\lambda = 1$ barely moves the large leaf, but cuts the small one by a third

Example: What λ does to the leaves

Credit data using only balance. The four leaves of round 1. Each account has $h_i = p(1-p) = 0.032$. Leaf values $w = -G/(H+\lambda)$:

leaf	H	$\lambda = 0$	$\lambda = 1$	$\lambda = 10$
$4{,}176$ accounts (15 defaults)	134.0	-0.92	-0.92	-0.86
604 accounts (54 defaults)	19.4	1.75	1.67	1.16
160 accounts (55 defaults)	5.1	9.68	8.10	3.28
60 accounts (42 defaults)	1.9	20.77	13.67	3.35

- $\lambda = 1$ barely moves the large leaf, but cuts the small one by a third
- the exact correction for the small leaf is 4.22: the Newton step overshoots most where H is small

Plan for today

• Recap: one round, and what XGBoost changes	10 min
• One leaf: its best correction	6 min
• One leaf or two? The gain	14 min
• Histograms: searching thresholds fast	12 min
• LightGBM	14 min
• In practice, and two questions	12 min

Recall: XGBoost's objective

1 2 3 4 5 6 box 5: set the leaf values

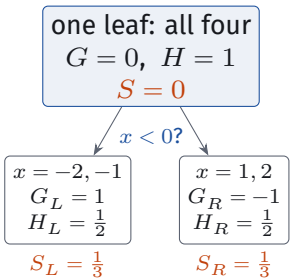
Proposition

For fixed leaves $R_1, \dots, R_T$ with every $H_j + \lambda > 0$, the approximate objective is smallest at

$$w_j^* = -\frac{G_j}{H_j + \lambda} \qquad \text{with value} \qquad \text{const} - \sum_{j=1}^{T} \frac{G_j^2}{2(H_j + \lambda)} + \gamma T.$$

XGBoost: Is it worth splitting this leaf?

Ex: the four points $x = (-2, -1, 1, 2)$, $y = (0, 0, 1, 1)$ in round 1:
$g = (\frac{1}{2}, \frac{1}{2}, -\frac{1}{2}, -\frac{1}{2})$, every $h_i = \frac{1}{4}$; $\lambda = 1$.



At its best value, a leaf with sums G, H lowers the objective by (the proposition)

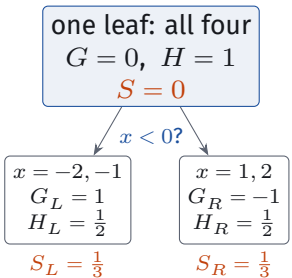
$$S(G, H) = \frac{G^2}{2(H + \lambda)}.$$

A split gives each child its own best value:

$$\boxed{\text{Gain} = S_L + S_R - S - \gamma}$$

XGBoost: Is it worth splitting this leaf?

Ex: the four points $x = (-2, -1, 1, 2)$, $y = (0, 0, 1, 1)$ in round 1: $g = (\frac{1}{2}, \frac{1}{2}, -\frac{1}{2}, -\frac{1}{2})$, every $h_i = \frac{1}{4}$; $\lambda = 1$.



At its best value, a leaf with sums G, H lowers the objective by (the proposition)

$$S(G, H) = \frac{G^2}{2(H + \lambda)}.$$

A split gives each child its own best value:

$$\boxed{\text{Gain} = S_L + S_R - S - \gamma}$$

Here $\frac{1}{3} + \frac{1}{3} - 0 - \gamma = \frac{2}{3} - \gamma$, so split if $\text{Gain} = \frac{2}{3} - \gamma > 0$.

The gain, written out

With $S(G, H) = G^2/(2(H + \lambda))$, and $G = G_L + G_R$, $H = H_L + H_R$:

$$\begin{aligned}\text{Gain} &= S_L + S_R - S - \gamma \\ &= \frac{1}{2}\left[\underbrace{\frac{G_L^2}{H_L + \lambda} + \frac{G_R^2}{H_R + \lambda}}_{\text{two leaves}} - \underbrace{\frac{(G_L + G_R)^2}{H_L + H_R + \lambda}}_{\text{one leaf}}\right] - \underbrace{\gamma}_{\text{a leaf's cost}}.\end{aligned}$$

- the gain needs only the four sums G_L, H_L, G_R, H_R

The gain, written out

With $S(G, H) = G^2/(2(H + \lambda))$, and $G = G_L + G_R$, $H = H_L + H_R$:

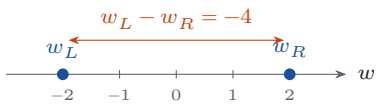
$$\begin{aligned}\text{Gain} &= S_L + S_R - S - \gamma \\ &= \frac{1}{2}\left[\underbrace{\frac{G_L^2}{H_L + \lambda} + \frac{G_R^2}{H_R + \lambda}}_{\text{two leaves}} - \underbrace{\frac{(G_L + G_R)^2}{H_L + H_R + \lambda}}_{\text{one leaf}}\right] - \underbrace{\gamma}_{\text{a leaf's cost}}.\end{aligned}$$

- the gain needs only the four sums G_L, H_L, G_R, H_R
- the tree is grown greedily: at each leaf, pick the threshold with the largest gain

What the gain measures

With $\lambda = 0$, and $w_L = -G_L/H_L$, $w_R = -G_R/H_R$ the children's best values,

$$\text{Gain} = \frac{1}{2} \underbrace{\frac{H_L H_R}{H_L + H_R}}_{\text{weight}} (w_L - w_R)^2 - \gamma.$$

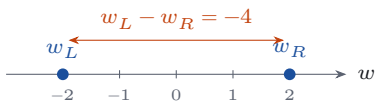


Ex: four points: weight $\frac{1}{4}$, gain $\frac{1}{2} \cdot \frac{1}{4} \cdot 16 = 2$

What the gain measures

With $\lambda = 0$, and $w_L = -G_L/H_L$, $w_R = -G_R/H_R$ the children's best values,

$$\text{Gain} = \frac{1}{2} \underbrace{\frac{H_L H_R}{H_L + H_R}}_{\text{weight}} (w_L - w_R)^2 - \gamma.$$



Ex: four points: weight $\frac{1}{4}$, gain $\frac{1}{2} \cdot \frac{1}{4} \cdot 16 = 2$

Key idea: Large when the new leaf values are far apart, and both have large enough H.

Three controls on a split

Ex: the four points: parent $G = 0$, $H = 1$; children $G = \pm 1$, $H = \frac{1}{2}$.

> **Recall:** $w = -G/(H+\lambda)$; $\text{Gain} = S_L + S_R - S - \gamma$ with $S = G^2/(2(H+\lambda))$. Here $S = 0$ and $S_L = S_R = 1/(1+2\lambda)$.

	asks	on the four points
λ	how big a correction?	w: $\mp 2 \to \mp\frac{2}{3}$; gain: $2 \to \frac{2}{3}$
γ	is one more leaf worth it?	split only if $\gamma < \frac{2}{3}$
`min_child_weight`	enough H in each child?	default $1 > H_L = \frac{1}{2}$: no split

Three controls on a split

Ex: the four points: parent $G = 0$, $H = 1$; children $G = \pm 1$, $H = \frac{1}{2}$.

Recall: $w = -G/(H+\lambda)$; $\text{Gain} = S_L + S_R - S - \gamma$ with $S = G^2/(2(H+\lambda))$. Here $S = 0$ and $S_L = S_R = 1/(1+2\lambda)$.

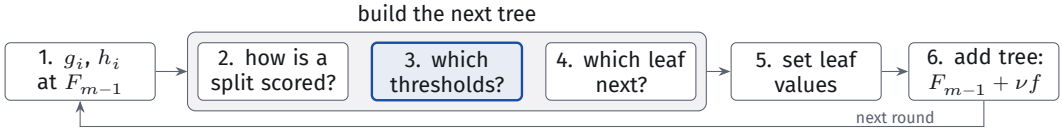
	asks	on the four points
λ	how big a correction?	w: $\mp 2 \to \mp\frac{2}{3}$; gain: $2 \to \frac{2}{3}$
γ	is one more leaf worth it?	split only if $\gamma < \frac{2}{3}$
`min_child_weight`	enough H in each child?	default $1 > H_L = \frac{1}{2}$: no split

Note: In the case of squared loss, every $h_i = 1$, so H is the leaf's number of points, which makes `min_child_weight` a minimum leaf size.

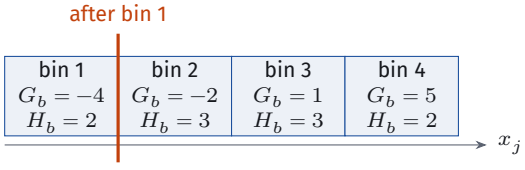
Plan for today

• Recap: one round, and what XGBoost changes	10 min
• One leaf: its best correction	6 min
• One leaf or two? The gain	14 min
• Histograms: searching thresholds fast	12 min
• LightGBM	14 min
• In practice, and two questions	12 min

A faster way to search thresholds: histograms



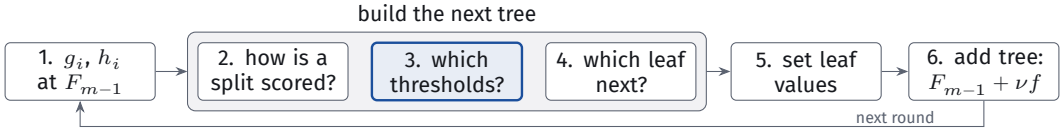
- bin each feature once; in each leaf, add each (g_i, h_i) into its bin
- move the cut one bin right: add that bin to the left; right $=$ total $-$ left



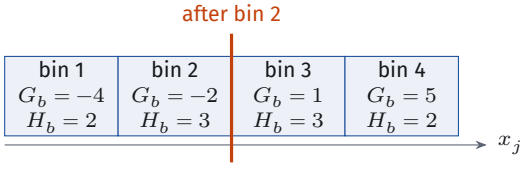
cut	G_L	H_L	G_R	H_R	gain
after 1	-4	2	4	8	5

$\lambda = 0$; the parent has $G = 0$

A faster way to search thresholds: histograms



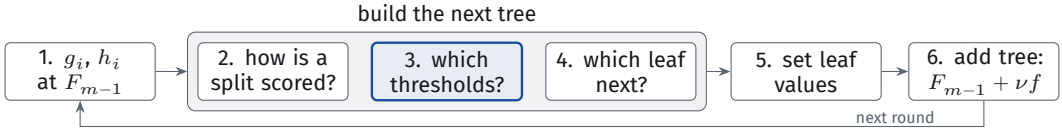
- bin each feature once; in each leaf, add each (g_i, h_i) into its bin
- move the cut one bin right: add that bin to the left; right $=$ total $-$ left



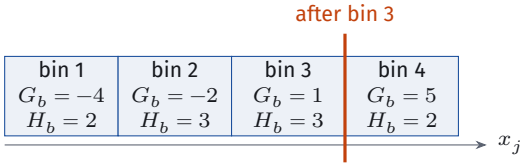
cut	G_L	H_L	G_R	H_R	gain
after 1	-4	2	4	8	5
after 2	-6	5	6	5	7.2

$\lambda = 0$; the parent has $G = 0$

A faster way to search thresholds: histograms



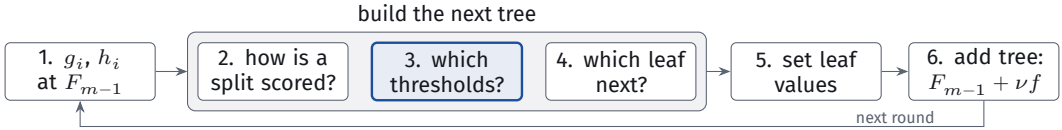
- bin each feature once; in each leaf, add each (g_i, h_i) into its bin
- move the cut one bin right: add that bin to the left; right $=$ total $-$ left



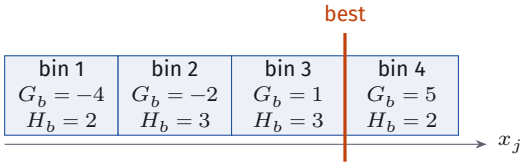
cut	G_L	H_L	G_R	H_R	gain
after 1	-4	2	4	8	5
after 2	-6	5	6	5	7.2
after 3	-5	8	5	2	7.8

$\lambda = 0$; the parent has $G = 0$

A faster way to search thresholds: histograms



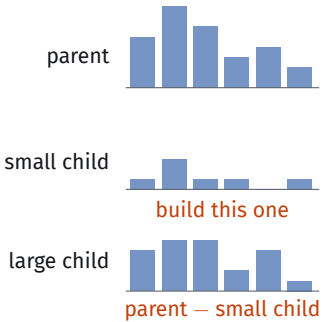
- bin each feature once; in each leaf, add each (g_i, h_i) into its bin
- move the cut one bin right: add that bin to the left; right $=$ total $-$ left
- only the set of thresholds is approximate; g_i, h_i and the gain are not



cut	G_L	H_L	G_R	H_R	gain
after 1	-4	2	4	8	5
after 2	-6	5	6	5	7.2
after 3	-5	8	5	2	7.8

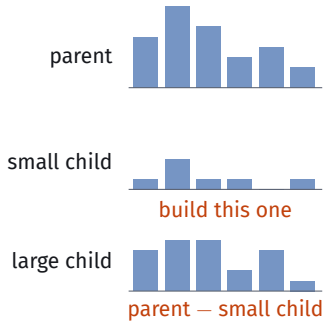
$\lambda = 0$; the parent has $G = 0$

Why histograms are fast



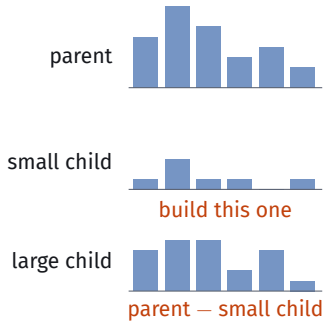
- exact: sort, then check every distinct value. Histogram: one pass of additions, then at most 255 cuts per feature

Why histograms are fast



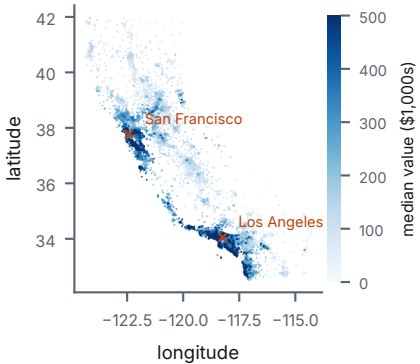
- exact: sort, then check every distinct value. Histogram: one pass of additions, then at most 255 cuts per feature
- subtraction: a child's histogram is its parent's minus its sibling's, so build only the smaller child's

Why histograms are fast



- exact: sort, then check every distinct value. Histogram: one pass of additions, then at most 255 cuts per feature
- subtraction: a child's histogram is its parent's minus its sibling's, so build only the smaller child's
- the bins are chosen once, before boosting starts, and reused in every leaf of every tree

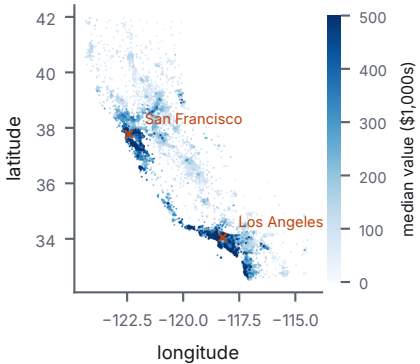
Example: California housing



A larger data set, our example for the rest of today: $20{,}640$ districts, 1990 census. Predict each district's median house value, capped at $\$500{,}001$ (4.8% at the cap).

- location, median age, rooms, bedrooms (207 missing), population, households, median income, and ocean proximity (5 categories)

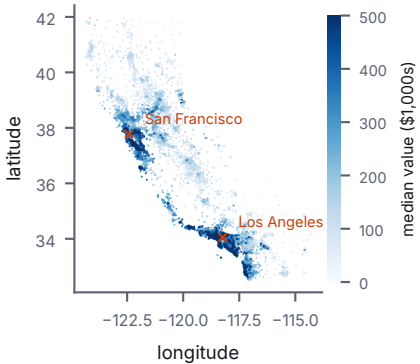
Example: California housing



A larger data set, our example for the rest of today: $20{,}640$ districts, 1990 census. Predict each district's median house value, capped at $\$500{,}001$ (4.8% at the cap).

- location, median age, rooms, bedrooms (207 missing), population, households, median income, and ocean proximity (5 categories)
- split districts into train (80%) and test (20%); validate on 20% of train; refit on all of train; test once on all test data

Example: California housing

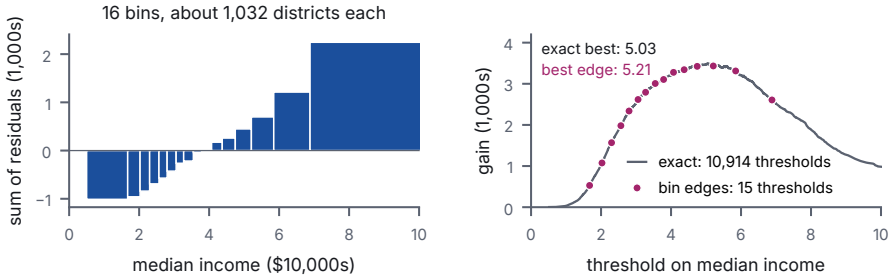


A larger data set, our example for the rest of today: $20{,}640$ districts, 1990 census. Predict each district's median house value, capped at $\$500{,}001$ (4.8% at the cap).

- location, median age, rooms, bedrooms (207 missing), population, households, median income, and ocean proximity (5 categories)
- split districts into train (80%) and test (20%); validate on 20% of train; refit on all of train; test once on all test data
- squared loss; error as RMSE, in dollars

Example: California's histograms

The first split on median income ($10{,}915$ distinct values), with 16 bins of equal counts. Left: each bin's sum of residuals, $-G_b$. Right: the gain at every cut.

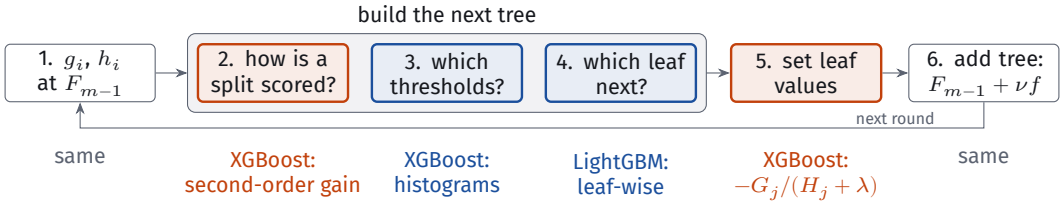


The best of 15 cuts keeps 98.4% of the best exact gain; with 256 bins (the default), 99.9%. 500 trees of depth 6: exact 3.5 s, histograms 0.6 s.

Plan for today

• Recap: one round, and what XGBoost changes	10 min
• One leaf: its best correction	6 min
• One leaf or two? The gain	14 min
• Histograms: searching thresholds fast	12 min
• LightGBM	14 min
• In practice, and two questions	12 min

LightGBM: the same engine

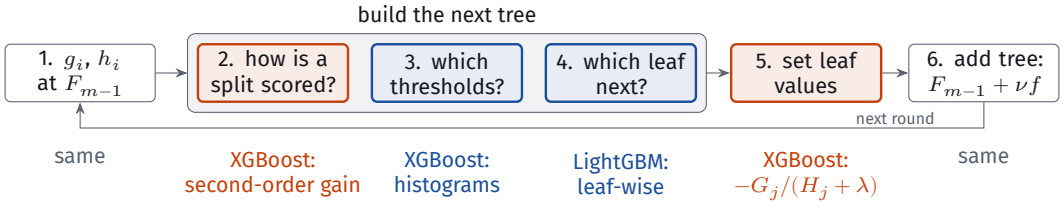


Definition (LightGBM (Ke et al., 2017))

Another fast implementation of gradient-boosted trees, with XGBoost's gain and leaf values.

- shared: boxes 2, 3, 5 (Ex: four points: same w, gain)

LightGBM: the same engine

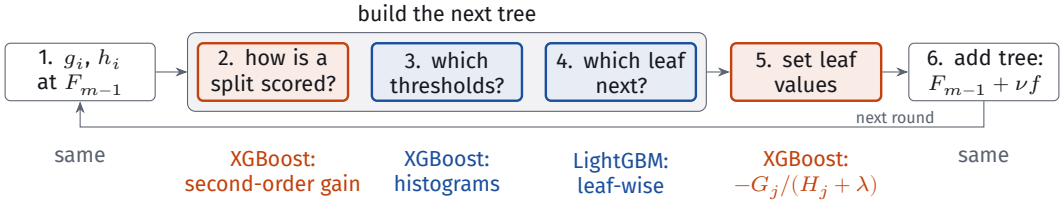


Definition (LightGBM (Ke et al., 2017))

Another fast implementation of gradient-boosted trees, with XGBoost's gain and leaf values.

- shared: boxes 2, 3, 5 (Ex: four points: same w, gain)
- characteristic: box 4, leaf-wise (XGBoost: depth-wise)

LightGBM: the same engine



Definition (LightGBM (Ke et al., 2017))

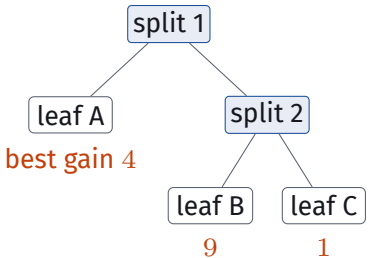
Another fast implementation of gradient-boosted trees, with XGBoost's gain and leaf values.

- shared: boxes 2, 3, 5 (Ex: four points: same w, gain)
- characteristic: box 4, leaf-wise (XGBoost: depth-wise)
- extras: other tricks for very large data

Box 4: one more split. Which leaf?



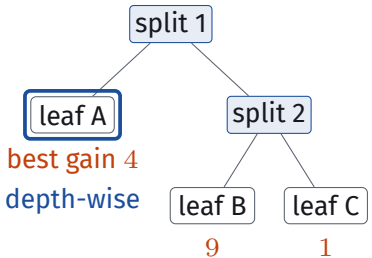
box 4: which leaf next?



Each leaf shows the gain of its best split.
One more split: which leaf?

Box 4: one more split. Which leaf?

1 2 3 4 5 6 box 4: which leaf next?



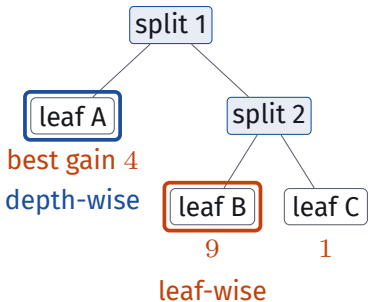
Each leaf shows the gain of its best split.
One more split: which leaf?

- depth-wise (XGBoost's default): the leaf nearest the root, A

Box 4: one more split. Which leaf?



box 4: which leaf next?



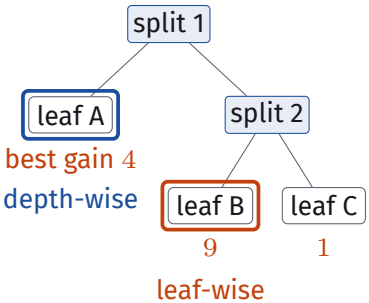
Each leaf shows the gain of its best split.
One more split: which leaf?

- depth-wise (XGBoost's default): the leaf nearest the root, A
- leaf-wise (LightGBM): the largest gain, B, wherever it is

Box 4: one more split. Which leaf?



box 4: which leaf next?



Each leaf shows the gain of its best split. One more split: which leaf?

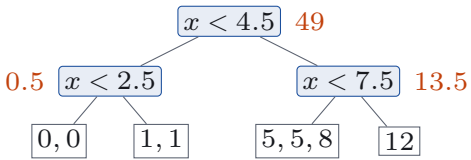
- depth-wise (XGBoost's default): the leaf nearest the root, A
- leaf-wise (LightGBM): the largest gain, B, wherever it is
- leaf-wise, repeated: deep, lopsided trees

Example: Eight points, grown two ways

$y = (0, 0, 1, 1, 5, 5, 8, 12)$ at $x = 1, \ldots, 8$; squared loss, $F_0 = 4$, $\lambda = \gamma = 0$; four leaves. Splits show their gains; leaves, their y values.

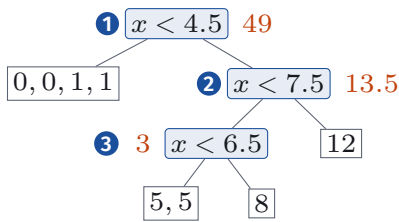
depth-wise, to depth 2

gradient boosting, XGBoost; RSS 6



leaf-wise, to 4 leaves

LightGBM, `lossguide`; RSS 1

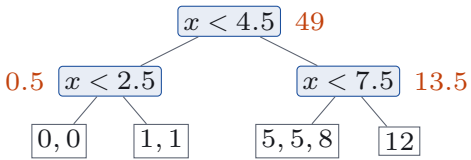


Example: Eight points, grown two ways

$y = (0, 0, 1, 1, 5, 5, 8, 12)$ at $x = 1, \ldots, 8$; squared loss, $F_0 = 4$, $\lambda = \gamma = 0$; four leaves. Splits show their gains; leaves, their y values.

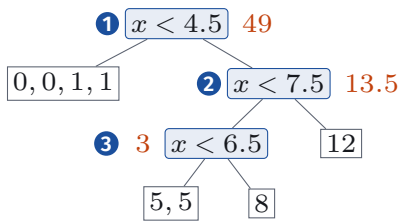
depth-wise, to depth 2

gradient boosting, XGBoost; RSS 6



leaf-wise, to 4 leaves

LightGBM, `lossguide`; RSS 1



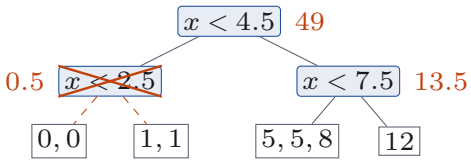
- leaf-wise: after split 1, the best gains are 0.5 and 13.5, so split 2 is on the right; then 0.5 against 3, so split 3 is on the right too

Example: Eight points, grown two ways

$y = (0, 0, 1, 1, 5, 5, 8, 12)$ at $x = 1, \ldots, 8$; squared loss, $F_0 = 4$, $\lambda = \gamma = 0$; four leaves. Splits show their gains; leaves, their y values.

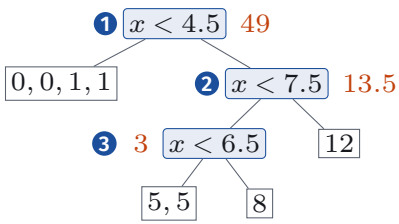
depth-wise, to depth 2

gradient boosting, XGBoost; RSS 6



leaf-wise, to 4 leaves

LightGBM, `lossguide`; RSS 1



- leaf-wise: after split 1, the best gains are 0.5 and 13.5, so split 2 is on the right; then 0.5 against 3, so split 3 is on the right too
- XGBoost with $\gamma = 1$: $0.5 - 1 < 0$, so that split is pruned

Why might leaf-wise growth overfit?

Question

Leaf-wise growth always makes the split that lowers the training loss most. Why might that hurt on new data?

Why might leaf-wise growth overfit?

Question

Leaf-wise growth always makes the split that lowers the training loss most. Why might that hurt on new data?

- it can keep refining one small, hard region, building a deep branch on a few points while the rest of the tree stays shallow

Why might leaf-wise growth overfit?

Question

Leaf-wise growth always makes the split that lowers the training loss most. Why might that hurt on new data?

- it can keep refining one small, hard region, building a deep branch on a few points while the rest of the tree stays shallow
- so limit it: `num_leaves`, `max_depth`, and the fewest points in a leaf, `min_data_in_leaf` (default 20)

Why might leaf-wise growth overfit?

Question

Leaf-wise growth always makes the split that lowers the training loss most. Why might that hurt on new data?

- it can keep refining one small, hard region, building a deep branch on a few points while the rest of the tree stays shallow
- so limit it: `num_leaves`, `max_depth`, and the fewest points in a leaf, `min_data_in_leaf` (default 20)

Why might leaf-wise growth overfit?

Question

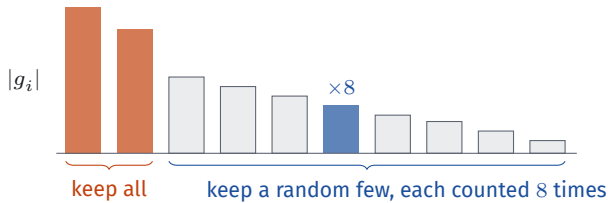
Leaf-wise growth always makes the split that lowers the training loss most. Why might that hurt on new data?

- it can keep refining one small, hard region, building a deep branch on a few points while the rest of the tree stays shallow
- so limit it: `num_leaves`, `max_depth`, and the fewest points in a leaf, `min_data_in_leaf` (default 20)

California, `num_leaves`	4	8	16	31	63	127
validation RMSE ($1,000s)	46.7	46.2	46.0	46.1	46.7	47.1

LightGBM's extras: GOSS

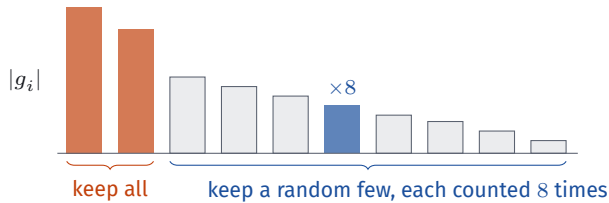
Gradient-based one-side sampling: build each tree from fewer points.



- large $|g_i|$: fit worst, and they drive the gains; keep them all

LightGBM's extras: GOSS

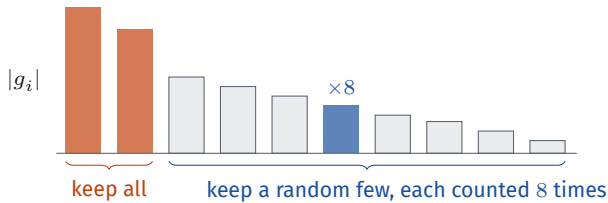
Gradient-based one-side sampling: build each tree from fewer points.



- large $|g_i|$: fit worst, and they drive the gains; keep them all
- the rest: a random sample, weighted so G and H stay about right

LightGBM's extras: GOSS

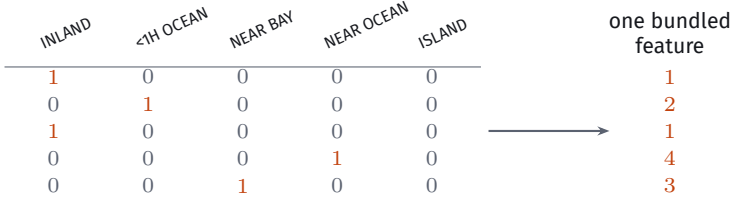
Gradient-based one-side sampling: build each tree from fewer points.



- large $|g_i|$: fit worst, and they drive the gains; keep them all
- the rest: a random sample, weighted so G and H stay about right
- an option; XGBoost: `sampling_method="gradient_based"`

LightGBM's extras: EFB

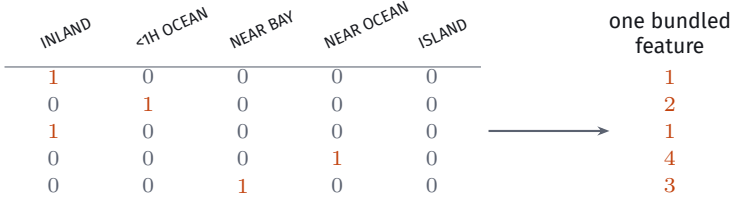
Exclusive feature bundling: features that are rarely nonzero in the same row share one histogram.



- exactly one 1 per district: one feature with five values, one histogram

LightGBM's extras: EFB

Exclusive feature bundling: features that are rarely nonzero in the same row share one histogram.

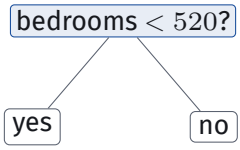


- exactly one 1 per district: one feature with five values, one histogram
- for thousands of sparse columns; XGBoost's search instead visits only the stored entries of a sparse matrix

Plan for today

- Recap: one round, and what XGBoost changes — 10 min
- One leaf: its best correction — 6 min
- One leaf or two? The gain — 14 min
- Histograms: searching thresholds fast — 12 min
- LightGBM — 14 min
- In practice, and two questions — 12 min

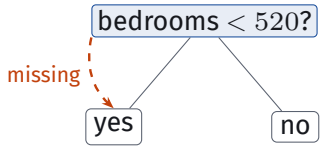
Missing values: a learned direction



California: 207 of the $20{,}640$ districts have no value for the number of bedrooms.

- at each split, try sending all the missing points left, then right, and keep the larger gain

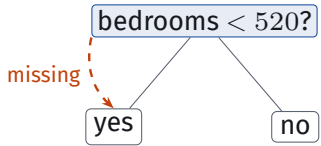
Missing values: a learned direction



California: 207 of the $20{,}640$ districts have no value for the number of bedrooms.

- at each split, try sending all the missing points left, then right, and keep the larger gain
- the direction is learned from the training data, split by split

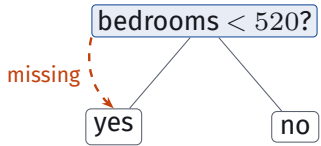
Missing values: a learned direction



California: 207 of the $20{,}640$ districts have no value for the number of bedrooms.

- at each split, try sending all the missing points left, then right, and keep the larger gain
- the direction is learned from the training data, split by split
- the fitted model: $2{,}092$ splits on bedrooms, missing sent left at 810

Missing values: a learned direction



California: 207 of the $20{,}640$ districts have no value for the number of bedrooms.

- at each split, try sending all the missing points left, then right, and keep the larger gain
- the direction is learned from the training data, split by split
- the fitted model: $2{,}092$ splits on bedrooms, missing sent left at 810
- both libraries do this; logistic regression (HW4 Q3) needs the value imputed

XGBoost and LightGBM: what differs

	XGBoost	LightGBM
objective	gain; $-G/(H+\lambda)$	the same
thresholds	histograms, or exact	histograms
growth	depth-wise; leaf-wise option	leaf-wise
fewer points	random, or gradient-based	random, or GOSS
sparse features	sparsity-aware search	EFB
missing values	a learned direction	the same

Key idea: Two implementations of one method. Same gradients, Hessians, gains and leaf values; different defaults for which leaf to split next, and different tricks for speed.

Example: California, tested once

Settings chosen on validation; refit on training; tested once on $4{,}128$ districts:

	test RMSE
linear regression (ocean proximity one-hot)	$\$68{,}700$
one tree (leaves ≥ 20; 638 leaves)	$\$56{,}000$
random forest (300 trees)	$\$47{,}700$
XGBoost, depth-wise, depth 4, $M = 1{,}414$	$\$44{,}200$
XGBoost, leaf-wise, 16 leaves, $M = 2{,}183$	$\$43{,}500$
LightGBM, leaf-wise, 16 leaves, $M = 871$	$\$43{,}800$

- within 1%; here the growth policy matters more than the library

Example: California, tested once

Settings chosen on validation; refit on training; tested once on $4{,}128$ districts:

	test RMSE
linear regression (ocean proximity one-hot)	$\$68{,}700$
one tree (leaves ≥ 20; 638 leaves)	$\$56{,}000$
random forest (300 trees)	$\$47{,}700$
XGBoost, depth-wise, depth 4, $M = 1{,}414$	$\$44{,}200$
XGBoost, leaf-wise, 16 leaves, $M = 2{,}183$	$\$43{,}500$
LightGBM, leaf-wise, 16 leaves, $M = 871$	$\$43{,}800$

- within 1%; here the growth policy matters more than the library
- on other data, hardware or versions the order can change

Python: XGBoost and LightGBM

```
from xgboost import XGBRegressor
from lightgbm import LGBMRegressor

xg = XGBRegressor(n_estimators=1414, max_depth=4,
                  learning_rate=0.1)        # depth-wise
lg = LGBMRegressor(n_estimators=871, num_leaves=16,
                   learning_rate=0.1)       # leaf-wise
xg.fit(X_train, y_train)
lg.fit(X_train, y_train)
# XGBoost, leaf-wise:
# XGBRegressor(grow_policy="lossguide", max_leaves=16,
#              max_depth=0, ...)
```

Two questions

Question (1)

Squared loss, $\lambda = \gamma = 0$, the same thresholds. Can L22's gradient boosting and XGBoost choose different splits?

Two questions

Question (1)

Squared loss, $\lambda = \gamma = 0$, the same thresholds. Can L22's gradient boosting and XGBoost choose different splits?

No: every $h_i = 1$, so the gain is half the RSS decrease.

Two questions

Question (1)

Squared loss, $\lambda = \gamma = 0$, the same thresholds. Can L22's gradient boosting and XGBoost choose different splits?

No: every $h_i = 1$, so the gain is half the RSS decrease.

Question (2)

The same g_i, h_i, λ, γ and thresholds. Can XGBoost and LightGBM build different trees?

Two questions

Question (1)

Squared loss, $\lambda = \gamma = 0$, the same thresholds. Can L22's gradient boosting and XGBoost choose different splits?

No: every $h_i = 1$, so the gain is half the RSS decrease.

Question (2)

The same g_i, h_i, λ, γ and thresholds. Can XGBoost and LightGBM build different trees?

Yes: depth-wise and leaf-wise growth spend a fixed number of leaves differently (the eight points). XGBoost's `lossguide` removes this difference.

Summary

- each round still adds one tree, times ν; what changes is how the tree is chosen
- XGBoost: one penalized second-order objective scores the splits, by the gain $S_L + S_R - S - \gamma$, and sets the leaf values $-G/(H + \lambda)$
- the gain is large when the children's best values are far apart; it can choose a different split than L22's RSS once the h_i differ
- scoring a split, choosing which thresholds to try, and choosing which leaf to split next are separate choices; histograms make the search fast in both libraries
- LightGBM: the same objective, grown leaf by leaf; GOSS and EFB for very large data