1. XGBoost for Regression

1. similarity score = $\dfrac{\text{sum of residuals squared}}{\text{number of residuals} + \lambda}$

   ↳ Regularization

* Note: "λ" is a regularization parameter & when $\lambda > 0$, it results in more pruning, by shrinking the similarity scores, and it results in smaller output values for the leaves.

2. Gain = Left similarity + Right similarity − Root similarity.

3. pruning: Gain − γ $\begin{cases} >0: \text{keep the node} \\ <0: \text{drop the node} \end{cases}$

2. XGBoost for classification:

$$\# \text{ similarity} = \frac{\left(\sum \text{Residuals}\right)^2}{\sum\left[\text{Prev. Probability}_i * (1 - \text{prev. Probability}_i)\right] + \lambda}$$

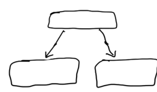
MATHS:

XGBoost uses loss functions to build trees by minimizing this equation:

$$\underbrace{\left[\sum_{i=1}^{n} L(y_i, P_i)\right]}_{\text{Loss function}} + \underbrace{\frac{1}{2}\lambda O_{value}^2}_{\text{Regularization}}$$

Regularization $\begin{cases} \lambda > 0; \text{ shrinks } O_{value} \\ \lambda < 0; \text{ expands } O_{value} \end{cases}$

→ The goal is to find the output value (O_{value}) for the leaf that minimizes the whole equation.

prediction = $\boxed{0.5}$ +  + other trees

(XGBoost tree)

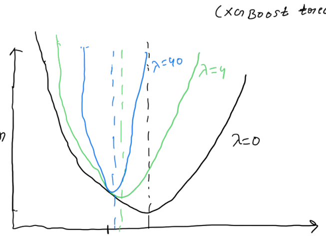


figure: O_{values} Vs loss function total for multiple regularization

* The more emphasis we give the regularization penalty by increasing "λ" the optimal O_{value} gets closer to 0.

XGBoost uses the second order Taylor approximation for both Regression & classification.

i.e.

$$L(y, P_i + O_{value}) \approx L(y, P_i) + \left[\frac{d}{dP_i} L(y, P_i)\right] O_{value} + \frac{1}{2}\left[\frac{d^2}{dP_i^2} L(y, P_i)\right] O_{value}^2$$

$$= \underset{\text{"Gradient"}}{L(y, P_i) + \underset{\downarrow}{g}\, O_{value}} + \frac{1}{2} \underset{\text{"Hessian"}}{\underset{\downarrow}{h}}\, O_{value}^2$$

Now, to minimize:

$$\left[\sum_{i=1}^{n} L(y_i, P_i^0 + O_{value})\right] + \frac{1}{2}\lambda O_{value}^2$$

to determine optimal O_{value}.

We can write it as:

$$\left.\begin{array}{l} \cancel{L(y_1, P_1^0)} + g_1 O_{value} + \frac{1}{2} h_1 O_{value}^2 + \\ \cancel{L(y_2, P_2^0)} + g_2 O_{value} + \frac{1}{2} h_2 O_{value}^2 + \cdots \cdots + \\ \cancel{L(y_n, P_n^0)} + g_n O_{value} + \frac{1}{2} h_n O_{value}^2 + \frac{1}{2} \lambda O_{value}^2 \end{array}\right\} \text{"Taylor Series Expansion"}$$

No contribution in O_{value}

so, for optimization we reduce it to:

$$(g_1 + g_2 + g_3 + \cdots \cdots + g_n) O_{value} + \frac{1}{2}(h_1 + h_2 + h_3 + \cdots \cdots + h_n + \lambda) O_{value}^2$$

$\frac{d}{dO_{value}}$

Now minimize the function by setting its derivative $= 0$.

i.e.

$$(g_1 + g_2 + g_3 + \cdots \cdots + g_n) + (h_1 + h_2 + h_3 + \cdots \cdots + h_n + \lambda) O_{value} = 0$$

or,

$$\boxed{O_{value} = \frac{-(g_1 + g_2 + g_3 + \cdots \cdots + g_n)}{(h_1 + h_2 + h_3 + \cdots \cdots + h_n + \lambda)}}$$

(A) For Regression:

* $L(y_i, P_i) = \frac{1}{2}(y_i - P_i)^2$

$\Rightarrow g_i = \frac{d}{dP_i} \frac{1}{2}(y_i - P_i)^2 = -(y_i - P_i) = -\text{Residual}$

lly, $h_i = \frac{d^2}{dP_i^2} \frac{1}{2}(y_i - P_i)^2 = \frac{d}{dP_i} - (y_i - P_i) = 1$

So, $\boxed{O_{value} = \frac{\text{sum of residuals}}{\text{Number of residuals} + \lambda}}$

This is the specific formula for the output value for a leaf when using XGBoost for regression.

(B) For Classification:

* $L(y_i, P_i) = -[y_i \log(P_i) + (1 - y_i)\log(1 - P_i)]$

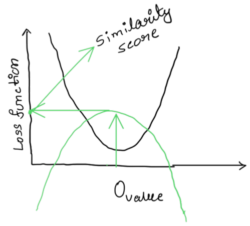
↳ $g_i = -(y_i - P_i) = -\text{Residuals}$

↳ $h_i = P_i * (1 - P_i)$

So, $$\boxed{O_{value} = \frac{\sum \text{Residuals}_i}{\sum[\text{Prev. Probability}_i * (1 - \text{Prev. Probability}_i)] + \lambda}}$$

(C) Similarity score for tree growing:

The first thing XGBoost does is multiply the optimization eqn by "-1" to flip the parabola upside down.



i.e.

$$-1 * \left[(g_1 + g_2 + \cdots \cdots + g_n) O_{value} + \frac{1}{2}(h_1 + h_2 + h_3 + \cdots \cdots + h_n + \lambda) O_{value}^2\right]$$

use the O_{value} from above & simplify:

$$\boxed{\text{Similarity Score} = \frac{1}{2} \frac{(g_1 + g_2 + \cdots \cdots + g_n)^2}{(h_1 + h_2 + \cdots \cdots + h_n + \lambda)}}$$

* For regression

$$\text{Similarity score} = \frac{\text{Sum of residuals, squared}}{\text{Number of residuals} + \lambda}$$

& cover = Number of residuals

And,

* For Classification

$$\text{Similarity score} = \frac{(\sum \text{Residuals}_i)^2}{\sum [\text{Prev. Probability}_i * (1 - \text{Prev. Probability}_i)] + \lambda}$$

$$\& \text{ cover} = \sum [P_i * (1 - P_i)]$$

Extra:

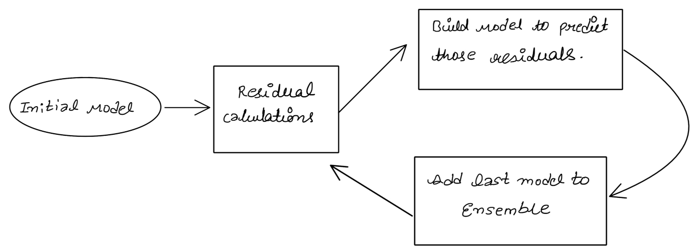


fig: General XGBoost Algorithm