GBDT: Decision Tree CART algo. $\begin{cases} \text{regression} \\ \text{classification.} \end{cases}$

$$f_M(x) = \sum_{m=1}^{M} f_m(x; \theta_m)$$

$\uparrow$ params

回归: MSE

分类: cross entropy - 0/1 is used to describe how well $f(x)$ approx true prob. dist. (when $\hat{f}(x)$ represents prob.)

exp loss - -1/1 $\exp\{-y_i f(x_i)\}$

custom loss fn generalization.

$$f_m(x) = f_{m-1}(x) + T(x; \theta_m)$$

$f_{m-1}(x)$ is the current fn. want to train the next tree m through minimizing empirical loss $\hat{\theta}_m = \underset{\theta_m}{\operatorname{argmin}} \sum_{i=1}^{N} L(y_i, f_{m-1}(x_i) + T(x_i; \theta_m))$

Binary Classification GBDT:

is simplified AdaBoost ① with $\alpha_m = 1 \ \forall m$

② 限定 each weak tree to be 二叉分类树

同时 update w_i, 只要 loss fn is exp. we can use exp to adjust weights for each tree.

Regression GBDT:

目标: $\mathcal{L}(y, f_m(x)) < \mathcal{L}(y, f_{m-1}(x))$

即:

$$\mathcal{L}(y, f_{m-1}(x)) - \mathcal{L}(y, f_m(x)) > 0$$

1st order Taylor Expansion $f(x) \approx f(x_0) + f'(x_0)(x - x_0)$

$f_m(x)$ is unknown and $f_m(x) = \underbrace{f_{m-1}(x)}_{\text{known}} + T(x; \theta_m)$

$\mathcal{L}(y, f(x))$ can be seen as a fn of $f(x)$ since y is deterministic.

$$\mathcal{L}(y, f_m(x))$$

$$\approx \mathcal{L}(y, f_{m-1}(x)) + \frac{\partial \mathcal{L}(y, f(x))}{\partial f(x)}\Big|_{f(x) = f_{m-1}(x)} \cdot (f_m(x) - f_{m-1}(x))$$

$$= \mathcal{L}(y, f_{m-1}(x)) + \frac{\partial \mathcal{L}(y, f(x))}{\partial f(x)}\Big|_{f(x) = f_{m-1}(x)} \cdot T(x; \theta_m)$$

$$\Rightarrow \underbrace{\mathcal{L}(y, f_{m-1}) - \mathcal{L}(y, f_m)}_{\text{Want } > 0} = -\frac{\partial \mathcal{L}}{\partial f}\Big|_{f = f_{m-1}} \cdot T(x; \theta_m)$$

当 $T(x; \theta_m) \approx -\frac{\partial \mathcal{L}}{\partial f}\Big|_{f = f_{m-1}}$ 时, we have RHS ≥ 0

$\Rightarrow$ 让 树 m $T(x; \theta_m)$ approx. 负梯度

$$r_m(x, y) = -\frac{\partial \mathcal{L}(y, f(x))}{\partial f(x)}\Big|_{f(x) = f_m(x)}$$

$\mathcal{L}(y, f(x))$ Loss fn can be custom

将 (x_i, y_i) 代入 $r_m(x, y)$，可得 r_{mi}

进而得到 m 轮 train set $T_m = \{(x_1, r_{m1}) \dots (x_N, r_{mN})\}$

Sum:
① compute negative gradient of current loss fn.
② construct new training set
③ train m weak tree on the set, get $T(x; \theta_m)$

GBDT Walk - Through:

input: $(x_1, y_1) \dots (x_N, y_N)$

output: regression tree $f(x)$

eg. $L(y, c) = \sum_i (y_i - c)^2$
$\Rightarrow c = \bar{y}$

① Initialization $f_0(x) = \underset{c}{\arg\min} \sum_{i=1}^{N} L(y_i, c)$ 常数

②. for $m \in [M]$:

a. for $i \in [N]$: $r_{m,i} = -\left[\frac{\partial L(y_i, f(x_i))}{\partial f(x_i)}\right]_{f(x) = f_{m-1}(x)}$

b. train weak tree m on r_m, get rectangles for leaf nodes $R_{m,j}$, $j \in [J]$

c. for $j \in [J]$: compute $c_{mj} = \underset{c}{\arg\min} \sum_{x_i \in R_{m,j}} L(y, f_{m-1}(x_i) + c)$

d. update

$$f_m(x) = f_{m-1}(x) + \sum_{j=1}^{J} c_{mj} \cdot \mathbb{1}(x \in R_{mj})$$

③ get regression tree $f(x) = f_M(x) = \sum_{m=1}^{M} \sum_{j=1}^{J} c_{mj} \cdot \mathbb{1}(x \in R_{mj})$

how each weak tree learn? 拟合负梯度

目标: $\mathcal{L}(y, f_{m-1}) - \mathcal{L}(y, f_m) > 0$

eg. MSE $\mathcal{L} = \frac{1}{2}(y - f_m(x))^2$

$$-\frac{\partial \mathcal{L}}{\partial f_m} = y - f_m(x) = r_m \quad \text{residual}$$

$\Rightarrow$ equiv to approx residual

so gradiant boosting can explain 一般性 MSE loss regression tree boosting

一般提升树: $\hat{\theta}_m = \arg\min_{\theta_m} \sum_{i=1}^{N} \mathcal{L}(y^{(i)}, f_{m-1}(x^{(i)}) + T(x^{(i)}; \theta_m))$

$$= \arg\min_{\theta_m} \sum_{i=1}^{N} (\underset{\text{residual}}{\underset{\uparrow}{r_m^{(i)}}} - T(x^{(i)}; \theta_m))^2$$