Boosting

Tuesday, December 7, 2021 1:12 PM

Q: Guess the word?

[bum + bum]

A: "Stymie"

Agenda:

1. Boosting Intuition
2. Boosting math.
3. Gradient Boosting
   - CBOT
4. Bias Variance.

Boosting Intuition.

Boosting: Base learner + algorithmic Corrective

"Hey, Base learner [x_1, x_2]" + "correct!" — algorithm that takes boosting base learner.

"Dog classifier" — base learner

CBOT

* Gradient Boosting

Core Idea: "Predict the Error"

Simplicity: "Fit → Predict residual"

Dataset $= \{x^{(i)}, y^{(i)}\}_{i=1}^{n}$

Step 0: Dataset → M_0 → hypothesis line that $\{x^{(i)} > 20\}$ $\{\beta_1 = 1\}$ $\{\beta_0 = 0\}$

$F_0 \Rightarrow \boxed{y = h(x)} \rightarrow \hat{y}$

"avg mean"

"Hope, hain't exist"

b) compute residuals.

$$\epsilon^{(i)} = \boxed{\underbrace{y^{(i)}}_{\text{actual}} - \underbrace{h_0(x^{(i)})}_{\text{prediction}}} \quad \left[\text{pseudo } \hat{y}\right] \text{ residuals}$$

for each datapoint $\{x^{(i)}, y^{(i)}, \epsilon^{(i)}\}_{i=1}^{n}$

Step 1: $\{\underbrace{x^{(i)}}_{\text{input feature}}, \underbrace{\epsilon^{(i)}}_{\text{output}}\}_{i=1}^{n} \longrightarrow M_1$

$M_1 = h_1(x)$

$h_1(x)$ — learnt hypothesis function

$F_1(x) =$ final model at the end of step 1.

$\boxed{F_1(x) = ?}$ $\quad h_0(x) + \alpha_1^* \cdot h_1(x)$

"best weight"

Step 2: $\{x^{(i)}, \epsilon^{(i)}\}_{i=1}^{n} \longrightarrow M_2$

$\{y^{(i)} - F_1(x^{(i)})\}$ $\quad h_2(x)$ learnt residuals

$\boxed{F_2(x) = h_0(x) + \alpha_1^* h_1(x) + \alpha_2^* h_2(x)}$

⋮

Step m: $F_m(x) = h_0(x) + \alpha_1^* h_1(x) + \alpha_2^* h_2(x) \ldots + \alpha_m^* h_m(x)$

$$\boxed{F_m(x) = h_0 + \sum_{i=1}^{m} \alpha_i^* \cdot h_i(x)}$$

Example

$h_0(x) = 0.7$

$\epsilon^{(i)} = y^{(i)} - F_0(x)$

$= 1.2 - 0.7$

$\Rightarrow 0.5$

$F_m(x) \Rightarrow h_0(x) + \alpha_1^* \cdot h_1(x) + \alpha_2^* \cdot h_2(x) \ldots + \alpha_m h_m(x)$

0.7 — 0.5 — 0.4 — 0.3

$\boxed{\alpha_i^* = 1}$

$h_0(x) = \bar{y}$ $\quad h_1(x)$

$F_1(x) = h_0(x) + \alpha_1^* h_1(x)$

$F_2(x) = h_0(x) + \alpha_1^* h_1(x) + \alpha_2^* h_2(x)$

$F_m(x) = h_0(x) + \sum_{i=1}^{m} \alpha_i^* \cdot h_i(x)$

* Geometrically Intuition

Linear Regression Vs. GBDT

$\hat{y} = w_0 + \sum_{i=1}^{d} w_i \cdot x_i$ — weight, coefficient

$w_i \rightarrow$ learnt from scratch

$F_m(x) = h_0(x) + \sum_{i=1}^{m} \alpha_i^* \cdot h_i(x)$

$\underbrace{h_1 \; h_2 \; h_3 \ldots h_m}_{\text{Sequential learning}}$

Tricks?

(from) →

$\hat{y} \Rightarrow h_0(x^{(i)}) + \alpha_1 \cdot h_1(x^{(i)}) + \alpha_2 \cdot h_2(x^{(i)})$

$1.5 + 0.25 \times 2.8 + 0.1 \times 1.5 + 0.5 \times 0.8$

$\hat{y} \Rightarrow$

$F_B(x) \Rightarrow \underbrace{h_0(x)}_{\text{mean}} + \underbrace{\alpha_1^* \cdot h_1(x)}_{\times 0.4} + \underbrace{\alpha_2^* \cdot h_2(x)}_{\times 0.3} + \underbrace{\alpha_3^* \cdot h_3(x)}_{\times 0.5}$

Residual, Pseudo Residuals, loss function.

$F_m(x) \Rightarrow h_0(x) + \alpha_1 h_1(x) + \alpha_2 h_2(x) \ldots + \alpha_m h_m(x)$

$\text{argmin}_{F_m(x)} \sum_{i=1}^{n} \left(y^{(i)} - F_m(x)\right)^2$

for training mod $\{x^{(i)}\}$ use $\{x^{(i)}, \epsilon^{(i)}\}$ → to minimize from prev mod

loss function.

$\epsilon^{(i)} = \sum_{i=1}^{n} \left(y^{(i)} - \underbrace{F_m(x^{(i)})}_{\hat{y}}\right)^2$ $\quad (y - \hat{y})^2$ ← Squared error

$\frac{\partial L}{\partial \hat{y}^{(i)}} \Rightarrow \frac{\partial \left[y^{(i)} - \hat{y}^{(i)}\right]^2}{\partial \hat{y}^{(i)}}$ $\quad \left[y - \underbrace{F_m(x^{(i)})}_{\hat{y}}\right]^2$

$\frac{\partial L}{\partial \hat{y}^{(i)}} = -2\left[y^{(i)} - \hat{y}^{(i)}\right]$

$\boxed{\frac{-\partial L}{\partial F_m(x^{(i)})}} \Rightarrow 2\left[y^{(i)} - F_m(x^{(i)})\right]$

residual

$-\frac{\partial L}{\partial \hat{y}}$ or neg. gradient

neg derivative of loss function $\Rightarrow$ (pseudo residuals)

$\boxed{\text{Pseudo residuals}}$

... on Model/Learner.

$h_j(x) \leftarrow \{x^{(i)}, \epsilon^{(i)}\}$ → $\epsilon^{(i)} \rightarrow \boxed{y^{(i)} - F_m(x^{(i)})}$

def get_pseudo(L, x)

$\frac{\partial L(y, F(x))}{\partial F(x)}$

$\epsilon^{(i)} \rightarrow$ (Pseudo residuals)

$\Rightarrow -\frac{\partial L}{\partial F_m(x^{(i)})}$

RF → minimize homogeneous ✗

GBDT → minimize large (x) ✗

Stage p:

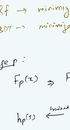
$F_p(x) \Rightarrow F_{p-1}(x) + \alpha \cdot h_p(x)$

$h_p(x) \xleftarrow{\text{fitted}} \{x^{(i)}, \epsilon^{(i)}\}$ → Add ✗ $\rightarrow F_p(x)$

$-\frac{\partial L}{\partial F_p(x^{(i)})}$

The Algorithm

(Pseudo code)



Input: training set $\{(x_i, y_i)\}_{i=1}^{n}$, a differentiable loss function $L(y, F(x))$, number of iterations M.

1. Initialize model with a constant value: $F_0(x) = \underset{\gamma}{\arg\min} \sum_{i=1}^{n} L(y_i, \gamma)$.
2. For $m = 1$ to M:
   1. Compute so-called pseudo-residuals: $r_{im} = -\left[\frac{\partial L(y_i, F(x_i))}{\partial F(x_i)}\right]_{F=F_{m-1}}$ for $i = 1, \ldots, n$.
   2. Fit a base learner (or weak learner, e.g. tree) $h_m(x)$ to pseudo-residuals, i.e. train it using the training set $\{(x_i, r_{im})\}_{i=1}^{n}$.
   3. Compute multiplier γ_m by solving the following one-dimensional optimization problem: $\gamma_m = \underset{\gamma}{\arg\min} \sum_{i=1}^{n} L(y_i, F_{m-1}(x_i) + \gamma h_m(x_i))$.
   4. Update the model: $F_m(x) = F_{m-1}(x) + \gamma_m h_m(x)$.
3. Output $F_M(x)$.

argmin $\frac{\partial L}{\partial \gamma}(y, \gamma)$

$\frac{\partial}{\partial \gamma}(y-\gamma)^2 + (y-\gamma)^2 = (y-\gamma)^2$

$2(y-\gamma) + 2(-1)(y-\gamma) = 0$

$4y - 4\gamma + 2\gamma - 2y = 0$

$4y + 2y = 3\gamma$

$\gamma = \frac{4y + 2y}{3} \Rightarrow \frac{y^{(1)} + y^{(2)} + y^{(3)}}{3}$

$\boxed{\gamma = \frac{\sum_{i=1}^{n} y^{(i)}}{n}} \rightarrow$ mean

Bias Variance Tradeoff

$F_p(x) = \sum_{i=1}^{p} \alpha_i^* \cdot h_i(x)$

* Bias Term?

Q: Good Bias? Higher Bias? Lower Bias?

Bias: m ↑ ↑ → model ↑ [x-y]

m ↓ ↓ → model ↓

Variance: m ↑ ↑ → model ↑ [x-y]

m ↓ ↓ → model ↓