⇒ Representing tree mathematically

$$T(x;\Theta) = \sum_{j=1}^{J} \gamma_j I(x \in R_j)$$

R_j – region

γ_j – prediction in region j

$$\Theta = \{R_j, \gamma_j\}_{j=1}^{J}$$

y_i – actual lable

x_i, y_i – traing pair

$$\hat{\Theta} = \underset{\Theta}{\operatorname{argmin}} \sum_{j=1}^{J} \sum_{x_i \in R_j} L(y_i, \gamma_j)$$

⇒ Boosted tree

$$f_M(x) = \sum_{m=1}^{M} T(x, \Theta_m)$$

$$\hat{\Theta}_m = \underset{\Theta}{\operatorname{argmin}} \sum_{i=1}^{N} L(y_i, f_{m-1}(x) + T(x_i, \Theta_m))$$

⇒ For a differentiable loss function

$$L(f) = \sum_{i=1}^{N} L(y_i, f(x_i))$$

N = no. of training examples

$$\hat{f} = \underset{f}{\operatorname{argmin}} L(f)$$

g_m = gradient at stage m

ρ_m = step-size (can be constant or using line search)

$$f_0 = h_0 \quad \text{(initial guess)}$$

$$f_M = \sum_{m=0}^{M} h_m \;; \quad h_m \in R^N$$

$$h_m = -\rho_m g_m$$

$$g_m = \left[\frac{\partial L(y_i, f(x_i))}{\partial f(x_i)}\right]_{f(x_i) = f_{m-1}(x_i)}$$

evaluating at $(m-1)$.

$$\rho_m = \underset{\rho}{\operatorname{argmin}} L(f_{m-1} - \rho \cdot g_m)$$

$$f_m = f_{m-1} - \rho_m g_m$$

⇒ g_m is unconstrained maximal descent direction.
- we want parametric form to simplify it
- we fit g_m using a tree.

$$\hat{\Theta}_m = \underset{\Theta}{\text{argmin}} \sum_{i=1}^{N} \underbrace{\left(-g_{im} - T(x_i;\Theta)\right)^2}_{\text{least square}}$$

- Above will be least square for any kind of original loss (L) (ranking-ndcg, classification algorithms)

original loss	gradient
$L = \frac{1}{2}(y_i - f(x_i))^2$	$g_m = y_i - f(x_i)$
$L = \lvert y_i - f(x_i) \rvert$	$\text{sign}(y_i - f(x_i))$
deviance	$I(y_i = G_k) - p_k(x_i)$

⇒ For given L, find g and then do regression with decision tree

$$\gamma_{jm} = \underset{\gamma}{\text{argmin}} \sum_{x_i \in R_j} L(y_i, f_{m-1}(x_i) + \gamma)$$

(earlier for simpler regression γ_{jm} were average residue error
- loss for Θ_m and γ_{jm} were same
- Now we are keeping them different)

- Might not be most ideal thing, but works well in practice