Lecture 8

- CART : Classification & Regression Trees

  - Recursive partitioning $(R_1, \dots, R_m, \dots, R_M)$

$\text{Loss}(R_1, \dots, R_m, \dots, R_M)$

purity :

- Classification :

$\{1, \dots, c, \dots C\}$ categories

in R_m, n_m = # examples in R_m

$n_{m,c}$ = # " " " in c

$$\hat{p}_{m,c} = \frac{n_{m,c}}{n_m}$$

- $$\text{Loss}(R_1, \dots, R_m, \dots, R_M) = -\sum_{m=1}^{M} n_m \sum_{c=1}^{C} \hat{p}_{m,c} \log p_{m,c}$$

$$= \sum_{m=1}^{M} n_m \, \text{entropy}(\hat{p}_m)$$

	1	2	$\cdots$	c	$\cdots$	C
$\hat{p}_m$	$\hat{p}_{m,1}$	$\hat{p}_{m,2}$	$\cdots$	$\hat{p}_{m,c}$	$\cdots$	$\hat{p}_{m,C}$

empirical distribution in R_m

- **Regression**:

$$\text{Loss}(R_1, \dots, R_m, \dots, R_M)$$

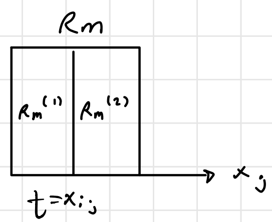
$$= \sum_{m=1}^{M} \sum_{i: x_i \in R_m} (y_i - \hat{c}_m)^2$$

$$\hat{c}_m = \text{average within } R_m$$

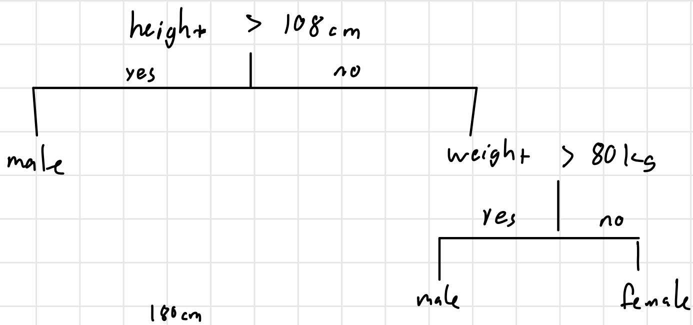
$$= \frac{\sum_{i: x_i \in R_m} y_i}{n_m}$$

- Recursive Partition:

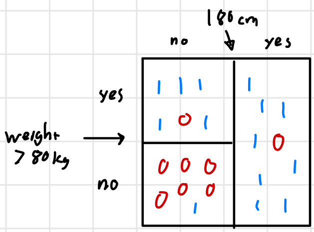
In step M $\{R_1, \dots, R_m, R_M$
    for m in $1:M$
        for j in $1:p$
            for t in $\{x_{ij},\ i = 1, \dots n\}$
                $\text{info\_gain} = \text{Loss}(R_1, \dots, R_m, \dots, R_M) - \text{Loss}(R_1, \dots, R_m^{(1)}, R_m^{(2)}, \dots, R_M)$



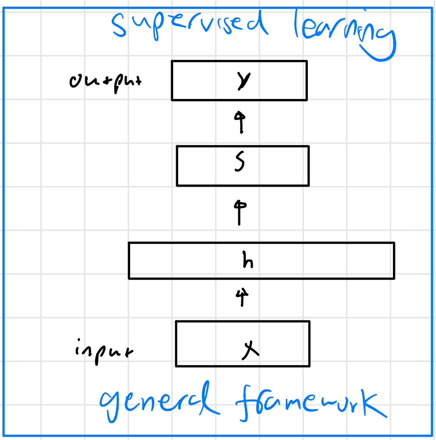
- Example:



Info gain



- Recall:



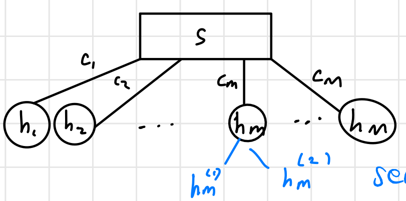
Regression:

$y \sim \mathcal{N}(s, \sigma^2 I)$

Classification:

$y \sim \text{softmax}(s)$

$$P(y=k \mid s) = \frac{e^{s_k}}{\sum_c e^{s_c}}$$



sequential expansion

$$\left.\begin{array}{l} \\ s = c_1 h_1 + \cdots + c_m h_m + \cdots c_M h_M \end{array}\right\} \text{One Tree}$$

$$h_m = \mathbb{1}(x \in R_m)$$

- We may express in matrix form:

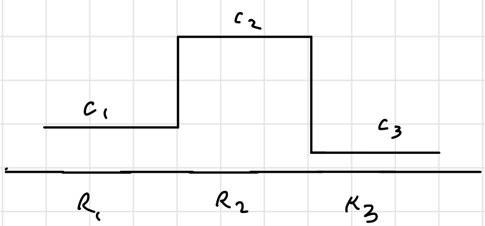
$$s = \begin{pmatrix} c_1 & \cdots & c_m & \cdots & c_M \end{pmatrix} \begin{pmatrix} h_1 \\ \vdots \\ h_m \\ \vdots \\ h_M \end{pmatrix}$$

$(= w\, h)$

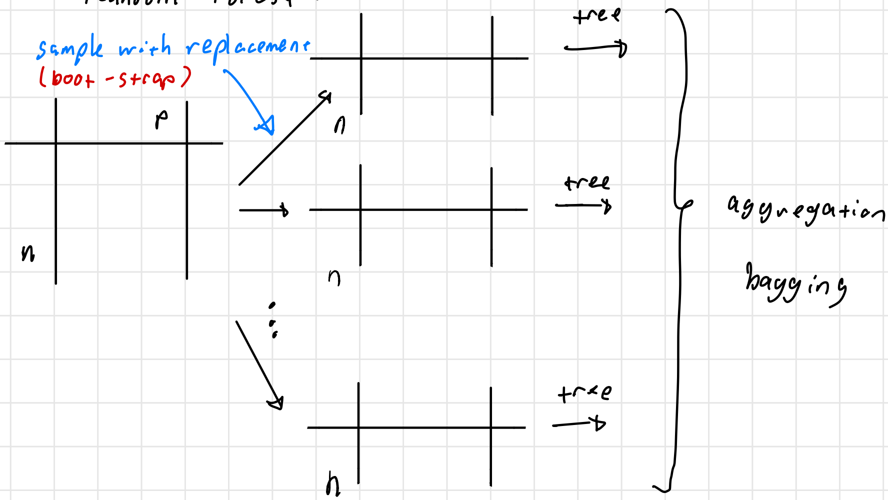
one-hot vector

- If $h_m = 1$, i.e $x \in R_m$ then $s = c_m$

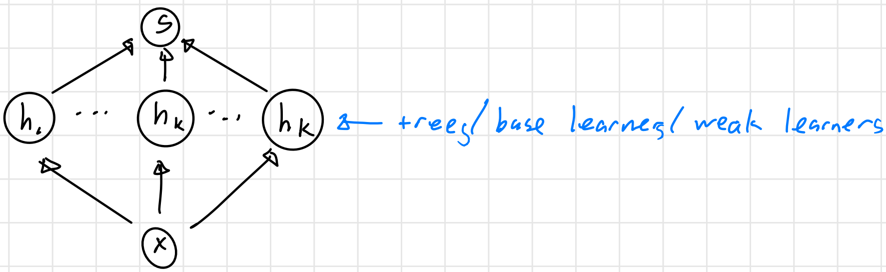
So $s = f(x) = c_m$ if $x \in R_m$



- Random Forest :



- Boosting



$$s = f(x) = \sum_{k=1}^{K} h_k(x)$$

- Extreme - Gradient - Boosting :

$$\text{Log-likelihood} = \sum_{i=1}^{n} \ell(s_i) = \sum_{i=1}^{n} \Big[\ell(\hat{s}_i) + \underbrace{\ell'(\hat{s}_i)}_{\hat{e}_i} \Delta s_i + \tfrac{1}{2} \underbrace{\ell''(\hat{s}_i)}_{-\hat{w}_i} \Delta s_i^2 \Big]$$

$$s_i = \underbrace{\sum_{k=1}^{t-1} h_k(x_i)}_{\hat{s}_i} + \underbrace{h_t(x_i)}_{\Delta s_i} \quad ?$$

$$= \text{const} + \tfrac{1}{2} \sum_{i=1}^{n} \hat{w}_i (\hat{y}_i - h_t(x))^2, \qquad \hat{y}_i = \frac{\hat{e}_i}{\hat{w}_i}$$

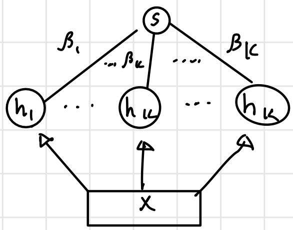
sequential addition

Epicycle

- Root : adaboost

$h_k(x) \in \{+1, -1\}$

weak classifiers



	$1 \ldots j \ldots p$	
1		
$\vdots$		
i	x_i^T	$y_i \in \{+1, -1\}$
$\vdots$		
n		

Theoretical Q:

weak learner = strong learner ?

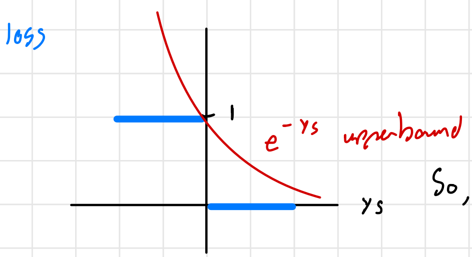
$$s = f(x) = \sum_{k=1}^{K} \underset{\text{voting right}}{\beta_k} h_k(x)$$

(v.s XGB, $h_k(x)$ = continuous output learned by weighted least squares)

regression tree

$$y = \text{sign}(s) = \begin{cases} +1 & s > 0 \\ -1 & s < 0 \end{cases}$$

- Exponential Loss Function :



if $ys \gg 0$ then we are very confident in the classification.

So, loss is $\mathbb{1}(y \neq \text{sign}(s)) \leq e^{-ys}$

continuous convex

encourage big margin ys

- So our loss function will be:

$$\text{Loss} = \sum_{i=1}^{n} e^{-y_i s_i}$$

boosting:

$$s = \underbrace{\sum_{k=1}^{t-1} \beta_k h_k(x_i)}_{\text{frozen}} + \underbrace{\beta_t h_t(x_i)}_{}$$

to be learned

$$\hat{s}_i \quad + \quad \Delta s_i$$

Thus $\text{Loss} = \sum_{i=1}^{n} e^{-y_i(\hat{s}_i + \Delta s_i)} = \sum_{i=1}^{n} \underbrace{e^{-y_i \hat{s}_i}}_{D_i} e^{-y_i \Delta s_i}$

$$D_i \leftarrow \frac{D_i}{\sum_{i=1}^{n} D_i}$$

$$\sum_{i=1}^{D} D_i = 1$$ Distribution Attention

If $-y_i \hat{s}_i$ is big, i recieves more attention

- Challenging examples:

$\text{sign}(y_i) = +1$ but $\hat{s}_i \ll 0$

$\text{sign}(y_i) = -1$ but $\hat{s}_i \gg 0$

- $\text{Loss} = \sum_{i=1}^{n} D_i e^{-y_i \Delta s_i} = \sum_{i=1}^{n} D_i e^{-y_i \beta_t h_t(x_i)}$

$$= \sum_{i=1}^{n} D_i e^{-\beta_t (\underbrace{y_i h_t(x_i)}_{})} = e^{-\beta_t} \underbrace{\sum_{i: y_i = h_t(x_i)} D_i}_{1-\varepsilon} + e^{\beta_t} \underbrace{\sum_{i: y_i \neq h_t(x_i)} D_i}_{\varepsilon \text{ error rate}}$$

y_i: +/−, $h_t(x_i)$: +/−

+1 if $y_i = h_t(x)$

−1 if $y_i \neq h_t(x)$

• Recall : $a+b \geq 2\sqrt{ab}$ because $(\sqrt{a}-\sqrt{b})^2 \geq 0$

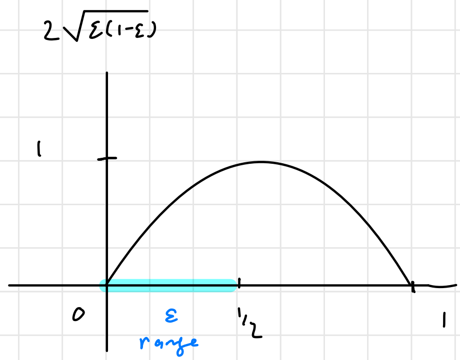
$$\text{Loss} \geq 2\sqrt{e^{-\beta_t}\sum_{i:y_i=h_t(x_i)} D_i \cdot e^{\beta_t}\sum_{i:y\neq h_t(x_i)} D_i} = 2\sqrt{\varepsilon(1-\varepsilon)}$$

min : $e^{-\beta_t}(1-\varepsilon) = e^{\beta_t}\varepsilon$

$$\frac{1-\varepsilon}{\varepsilon} = e^{2\beta_t}$$

So $\hat{\beta}_t = \frac{1}{2}\log\frac{1-\varepsilon}{\varepsilon}$

adaptive

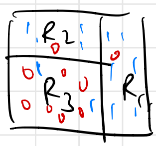


We choose $h_t(\ )$ to minimize ε

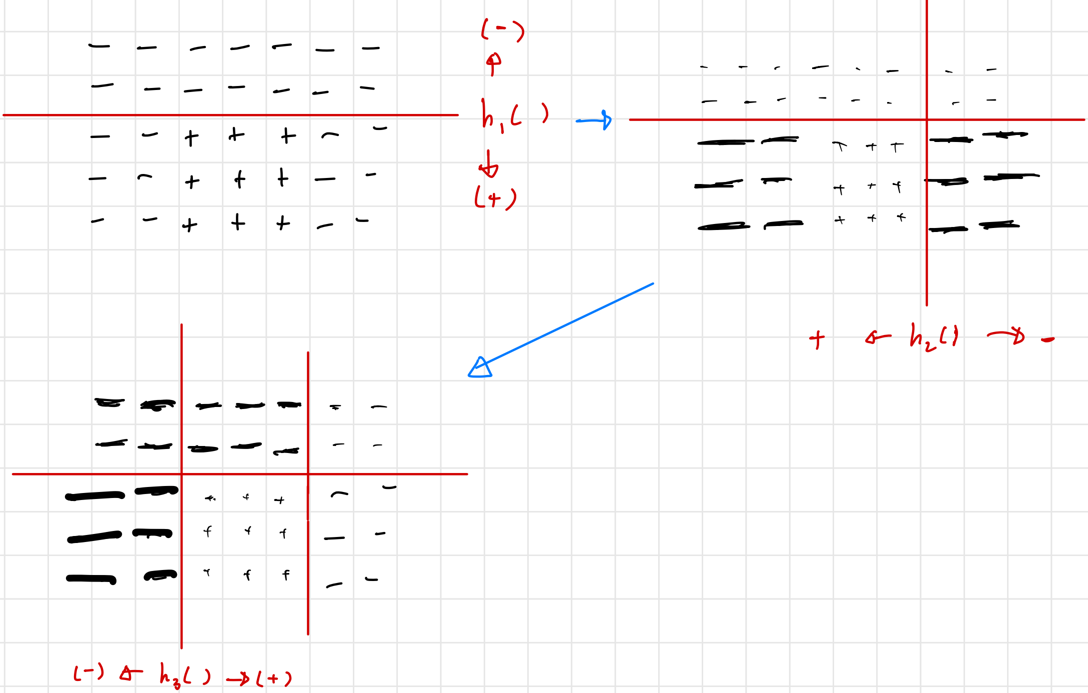
$$\varepsilon = \sum_{i:y_i\neq h_t(x_i)} D_i$$

Loss guiding

recursive partitioning



weak classifier : one layer tree



Keep adding trees, tends not to overfit

increasing margins

smoothing boundaries

XGB	vs	Adaboost
regression trees $\in \mathbb{R}$		classification trees $\in \{+1, -1\}$
general loss function		exponential loss function
2^{nd} order Taylor		closed form
fit to $\hat{e}_i$ with $\hat{w}_i$		$D_i \propto e^{-y_i \hat{S}_i}$

learn from error

error → new tree vs error back-prop