Ensembles

↳ group of musicians

ML: multiple models used together

$M_1, M_2, M_3 \cdots M_k$ ⟶ base models

↓ (Combine) more powerful model

(M)

4 Types

1. Bagging (Bootstrapped Aggregation) ⟶ very useful in real world
2. Boosting ⟶ high performing
3. Stacking ⟶ very powerful
4. cascading ⟶ kaggle – most

key aspect $M_1, M_2, M_3 \cdots M_k$

more different these models are, the better you can combine them.

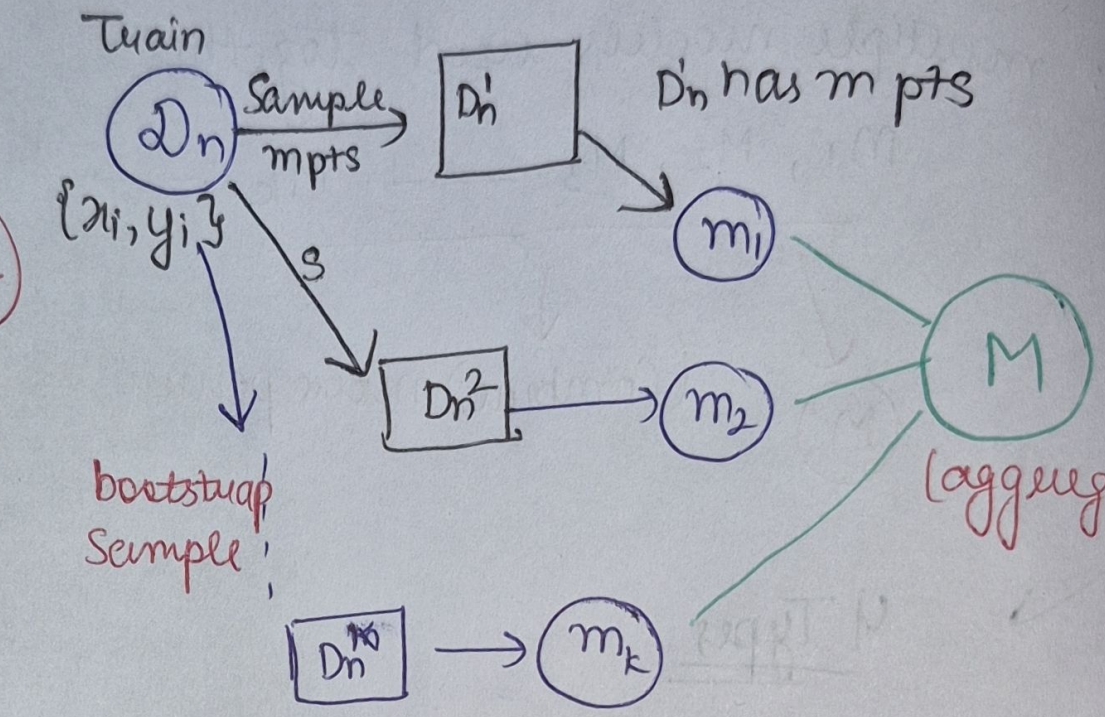
Problem: M_1: expert

Bagging (Bootstrap Aggregation)

↳ Statistics

(Random Forest)

Intuition



(Sampling with replacement)

m_i is built using D_n' of size m ($m \leq n$)

⇒ Each model m_i has seen different subset of data

Aggregation: Classification : Majority vote
Regression : mean / median

Bagging → can reduce variance in model without impacting the bias

model error = $Bias^2 + Var$

base model (M_i); low bias, high var model

Bagging (M_i's) → low bias; reduced variance

Bagging

bunch of low bias, high var models

→ DT of depth
↳ high var
low bias
↳ Random Forest

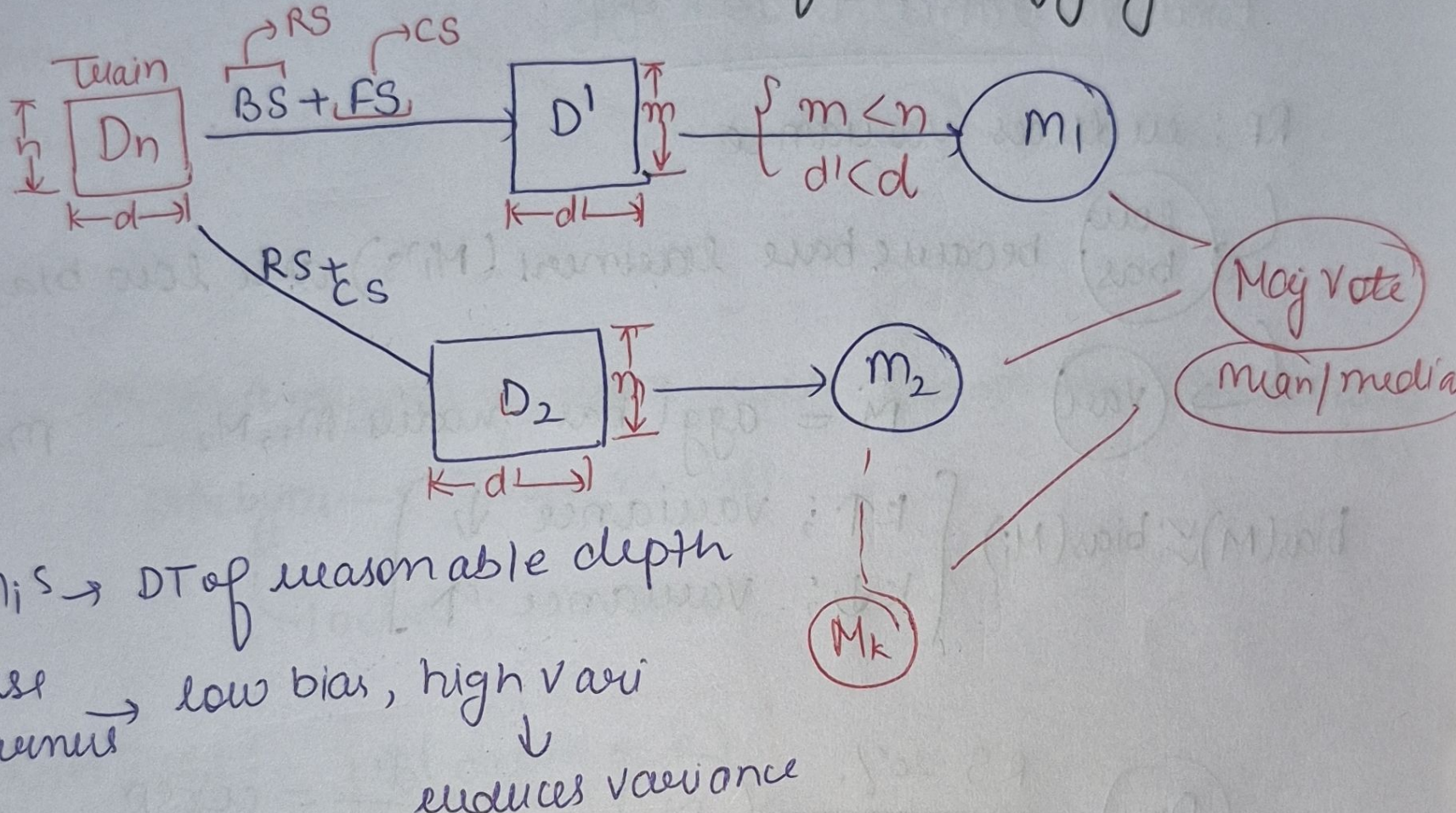
Combine them using bagging
↓
(low bias, reduced var)

Random Forest (Bagging)

- Decision Trees
- Random Bootstrap Sampling

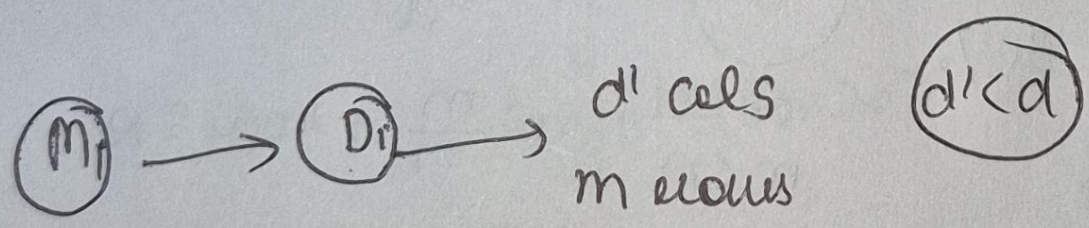
RF: DT + Bagging + Col. Sampling

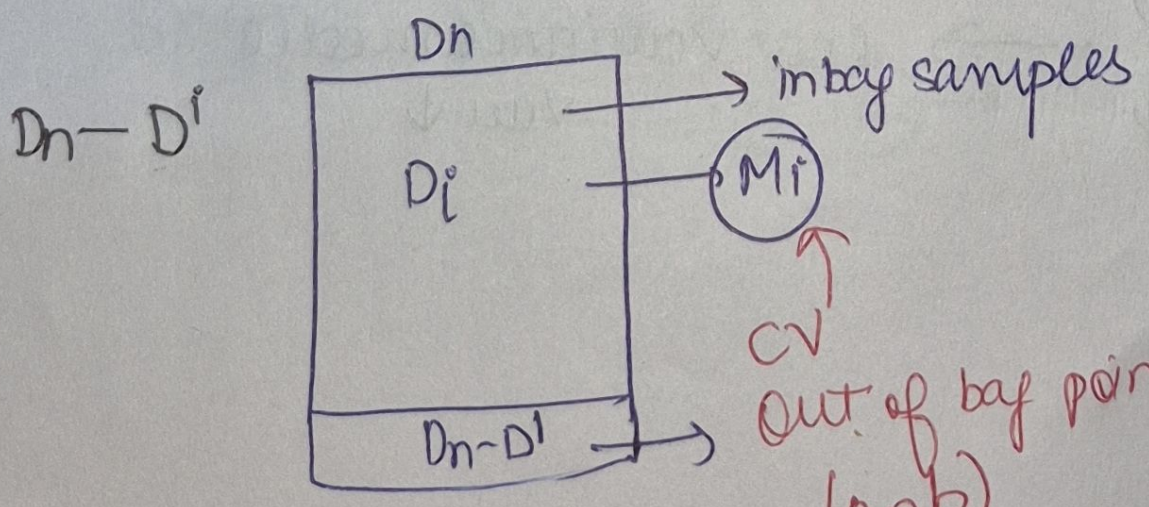
- DT ↑ base
- Bagging ↓ RS with replac
- Col. Sampling ↓ feature bagging



m_is → DT of reasonable depth

base learners → low bias, high vari → reduces variance





$M_i \rightarrow D^i$

$D_n - D_i \rightarrow$ oob samples = CV - dataset m_i

RF: DT base learners
+ row sampling with replacement
+ col sampling
+ agg (majority vote)
↳ (mean/median)

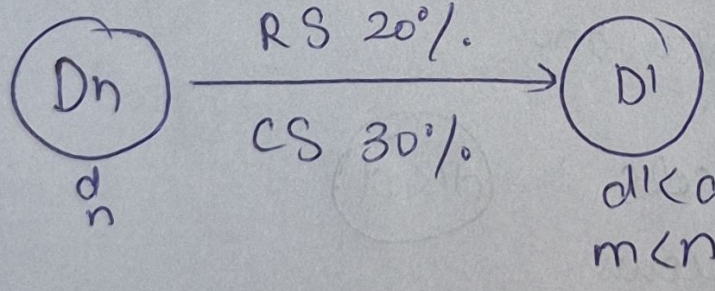
Bias Variance trade off

RF: reduces variance

↳ (low bias) because base learners (M_i's) are low bias

↳ (var) $\hat{M} = agg(\text{base models } M_1, M_2 \dots M_k)$

$bias(M) \approx bias(M_i)$ [RT; variance ↓ / K↓; variance ↑]



$\frac{d'}{d}$ = col SR

$\frac{m}{n}$ = row S.R

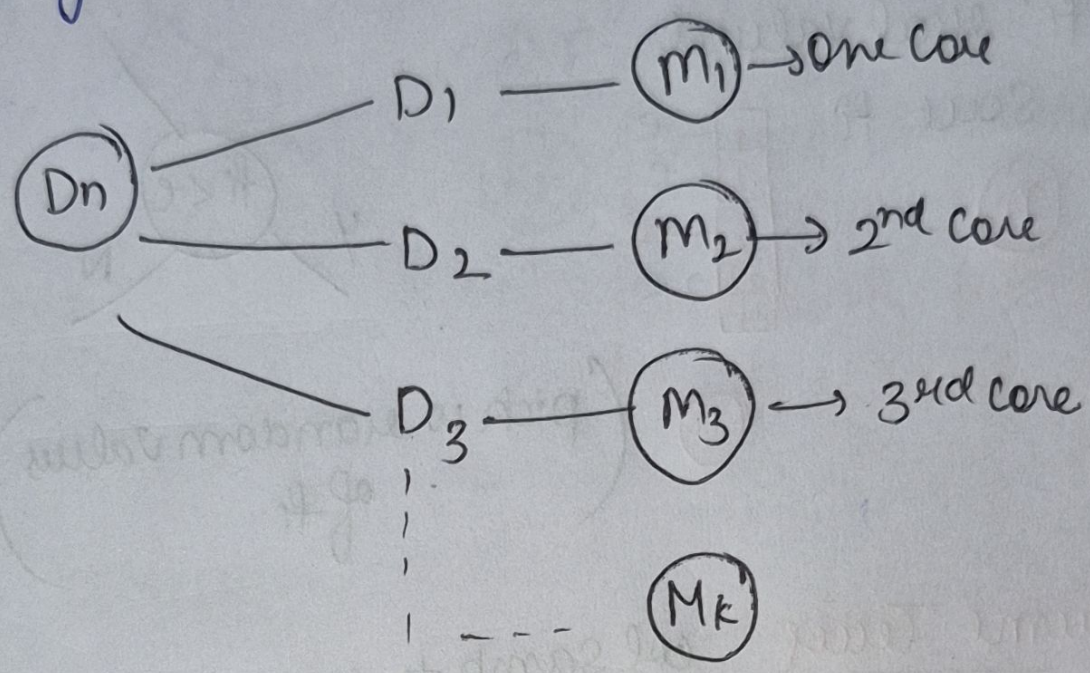
col. SR ↓ ⟶ low variance models
row SR ↓ var ↓

Train And Run Time Complexity

RF with K base learners (DT)

Train: $O(n \log d * K)$

multi core



Trivially Parallelizable

large amounts of data with reasonable # features (d)

peta bytes → Train models

↳ 100 bcx; 1000 bcx

Run Time: $O(depth * K)$

↓

large (10 to 20)

Space: $O(DT * K)$

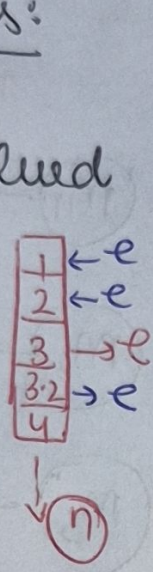
↳ if else → takes very less time & space

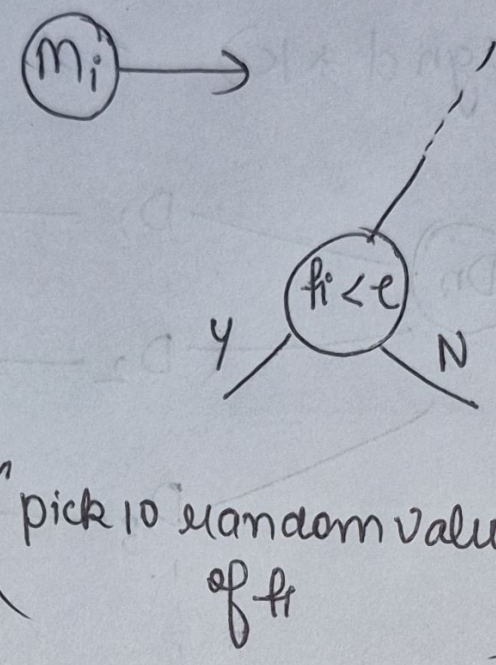
Extremely Randomized Trees → Try out random sam of possible value determine

RF → col-sampling, row sampling, agg
  └ reg DT

Extreme Trees:

RF/DT { fi: real valued
  sort fi
  (n)





(pick 10 random values of fi)

DT/ RF
↓
Try all possible values of fi
determine e

Extreme Trees ← col samp + row samp + agg
+ randomization when selecting "e"

Randomization as a way to reduce variance

RF → CS & R.S

ET → CS + R.S + randomization (-e)

(reduce variance better than RF)

Cons:

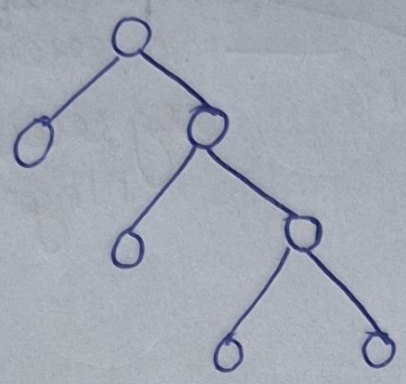
RF: DT + RS + CS + agg (reduce variance)

- DT → large dim; cate features with many categories
- do not handle
- cons for DT
- ① Bias Variance of RF ∪ K # base learners
  └ dup trees

(2) F

(2) Feature Imp

DT ; f_i :— overall reduction in entropy or gini Impurity because of this fea at various levels in DT



(RF) : overall reduction in H or I_G because of f_i @ various levels of each of M's

Boosting Intuition

Bagging : high var, low bias + reduce variance

base models

↓

Randomization + agg (CS, RS)

Boosting : low var, high bias + additively combine (GB) +

↓

reduce bias while keeping var low

error = $bias^2$ + Var + ε

Core Idea how Boosting reduce bias

clasfi / Regress

$D_{train} = \{x_i, y_i\}_{i=1}^{n}$

(0) D_{train} $\{x_i, y_i\}_{i=1}^{n}$ → M_0

high train error

↑

high bias, low var

{eg: DT which is shallow} low depth

Residu

(0) $D_{train} \rightarrow (M_0)$ high bias $\Rightarrow$ large train error

$(x_i, y_i)_{i=1}^n$ $\quad y = h_0(x)$

(b) $y_i - h_0(x_i) = error_i$ ← Simple diff error / Sq. error / log error / hinge error

$\{x_i, y_i, error_i\}_{i=1}^n$

(1) $(M_1) \leftarrow \{x_i, error_i\}_{i=1}^n$

$h_1(x)$

$F_1(x) =$ model at end of Stage 1.

$F_1(x) = \alpha_0 h_0(x) + \alpha_1 h_1(x)$] weighted sum of 2 base models

(2) $\{x_i, err_i\} \rightarrow (M_2)\ h_2(x)$

$\hookrightarrow y_i - F_1(x_i)$

$F_2(x) = \alpha_0 h_0(x) + \alpha_2 h_2(x) + \alpha_1 h_1(x)$

end of Stage K:

$$F_K(x) = \sum_{i=0}^{K} \alpha_i h_i(x)$$

→ trained to fit residual error @ end of the previous stage

additive weighted model

$h_i(x) \leftarrow \{x_i, err_i\}$

↑ residual error @ end of Stage (i-1)

Train err ↓ → bias ↓

$$F_K(x) = \sum_{i=0}^{K} \alpha_i h_i(x)$$

ends up having a low resid err

Gradient Boosted → DT

Residuals, Loss func & Gradients → Gradient Boosting

$$F_k(x) = \sum_{i=0}^{k} \alpha_i h_i(x)$$

Residual

@ end of stage k

$$res_i = y_i - F_k(x)$$

residual

$M_{k+1} \leftarrow \{x_i, res_i\}$

Loss Minimization

Logistic loss ← log reg / classfn
del Reg ← sq. loss
SVM ← hinge loss

regression

$$\mathcal{L}(y_i, F_k(x_i)) = (y_i - F_k(x))^2$$

↙

Sq loss

$$\frac{\partial \mathcal{L}}{\partial F_k(x_i)} = \frac{\partial \mathcal{L}}{\partial z_i}$$

$$\left[\begin{array}{l} \text{let} \\ F_k(x_i) = z_i \end{array}\right]$$

$$= \frac{\partial (y_i - z_i)^2}{\partial z_i}$$

$$= (-1) * 2(y_i - z_i)$$

$$\frac{\partial \mathcal{L}}{\partial F_k(x_i)} = -2(y_i - z_i)$$

$$-\frac{\partial \mathcal{L}}{\partial F_k(x_i)} = 2(y_i - F_k(x_i))$$

negative derivative

residual

(*) neg. gradient ≒ residual

pseudo residual

gradient boosting

$h_i(x)$ ⟵ $x_i, res_i = y_i - F_{i-1}(x)$

res ⟶ residual;

res_i ⟶ pseudo residual

$-\frac{\partial L}{\partial F_{i-1}(x)}$

Let us have any loss fn which is differentiable

RF ⟶

GB ⟶ min (any loss) ⟶ which is differentiable

↳ Super powerful

Stage i

$f_{i-1}(x)$

(x_i, res_i) ⟶ M_i ⟶ $h_i(x)$

↓

pseudo residual

$-\frac{\partial L}{\partial F_{i-1}(x)}$

Ques Is pseudo-residual $(-\partial y / \partial F_2(x))$ same as residual for non-squared loss loss func?

ans

ans

α $\frac{-\partial L}{\partial F_k(x)}$ ⇝ residuals for non squared loss func.

$$\frac{-\partial L}{\partial F_k(x)} = \alpha(y_i - F_k(x_i))$$

$\mathcal{L}$:- logg loss

$$\frac{\partial \mathcal{L}}{\partial F_k(x)} \qquad F_k(x) = \hat{y_i}$$

$$= p_i - y_i \quad \text{where} \quad p_i = \frac{1}{1+\exp(-\hat{y_i})}$$

$$= \frac{1}{1+\exp(-F_k(x))}$$

p_i = prob of $x_i \in$ class 1

pseudo residual $= \frac{-\partial L}{\partial F_k(x)} = p_i - y_i$

$y_i = 0$ or 1

$P(x_i \in 1)$

Gradient Boosting Regularizations & Shrinkage

$$F_m(x) = h_0(x) + \sum_{m=1}^{M} \gamma_m h_m(x)$$

M: # base models → hyperparameter

base models ↑ ⇒ overfit ↑ ⇒ Var ↑

C.V

M ↑ ↪ bias ↓ but could Var ↑

Shrinkage

Shrinkage

$$F_m(x) = F_{m-1}(x) + \nu \cdot \gamma_m h_m(x), \quad 0 < \nu \leq 1, \quad \nu \approx 0.1$$

ν is small
↳ 0.0001

ν ↓ overfitting ↓ Var ↓ [ν ↑ ⇒ overfitting ↑]

Train & Run Time Complexity

Train: $O(n \lg n d * M)$ M: # base learners

GBDT take more time to Train than RF

Run Time

GBDT: $O(depth * M)$ → Small in GB

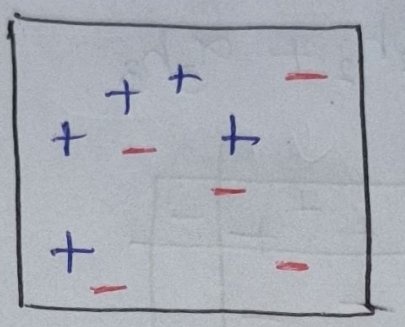
$O(depth * M + M) \approx O(depth * M)$

$h_m(x)$

Space $O(\text{store each tree} + \gamma m)$

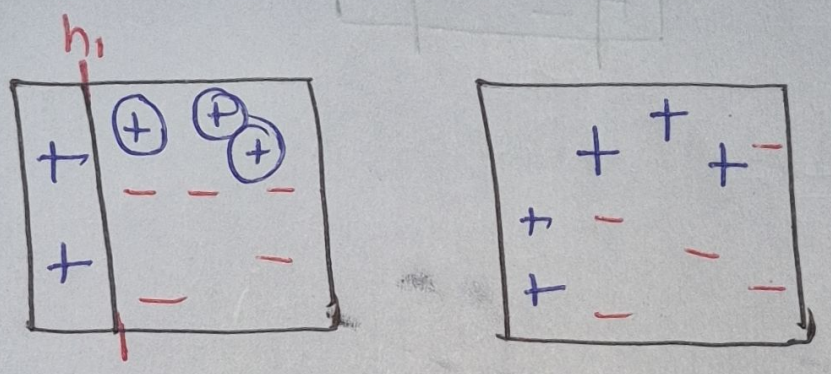
↳ if use

Adaa Boosting → image processing (face detection)



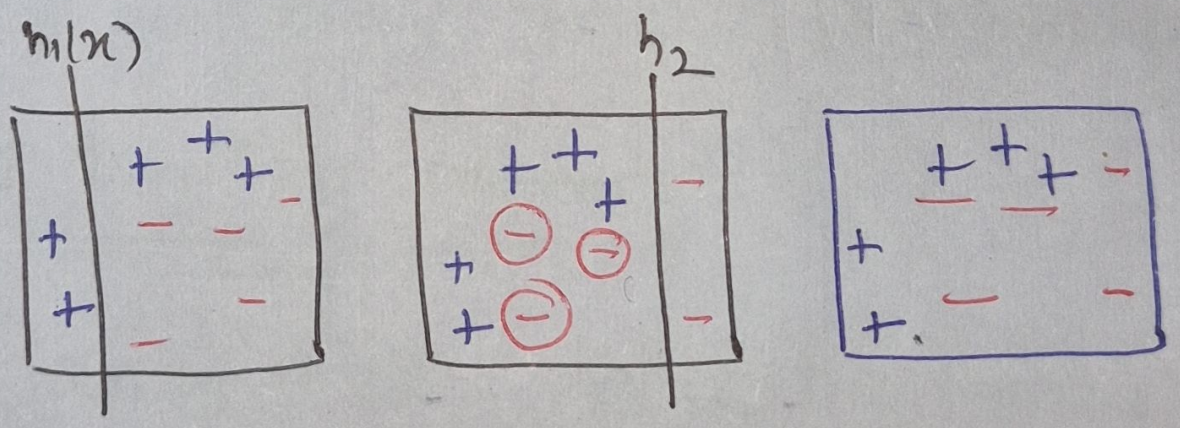
1st Round

$\alpha_1 h_1(x)$

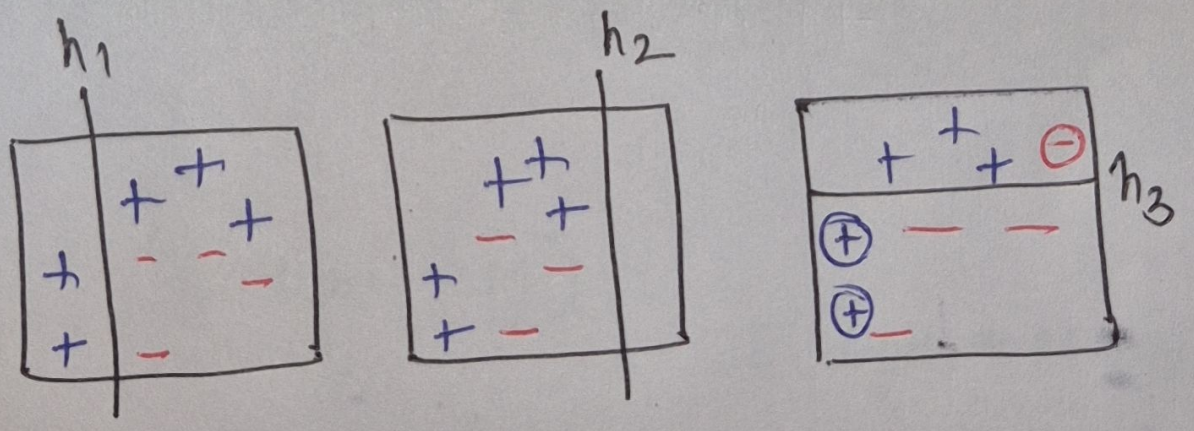


2nd Round

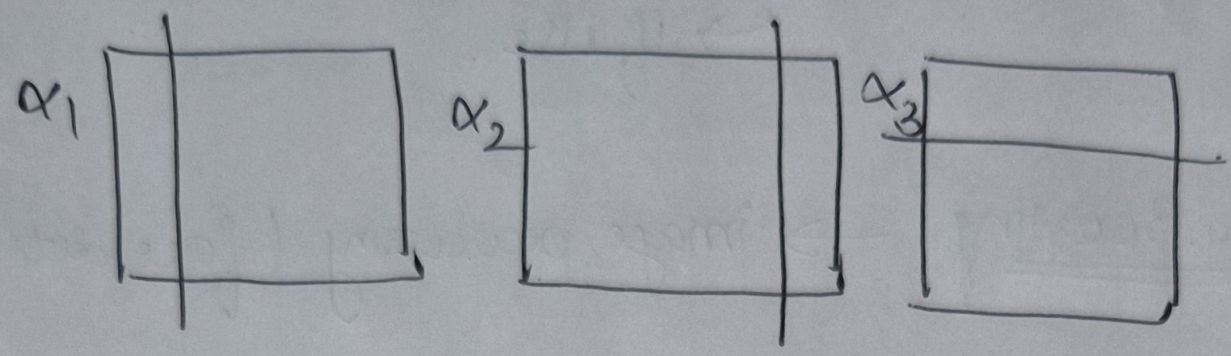
$\alpha_2 h_2$



3rd Round



final Model



$$F_3(x) = \alpha_1 h_1 + \alpha_2 h_2 + \alpha_3 h_3$$