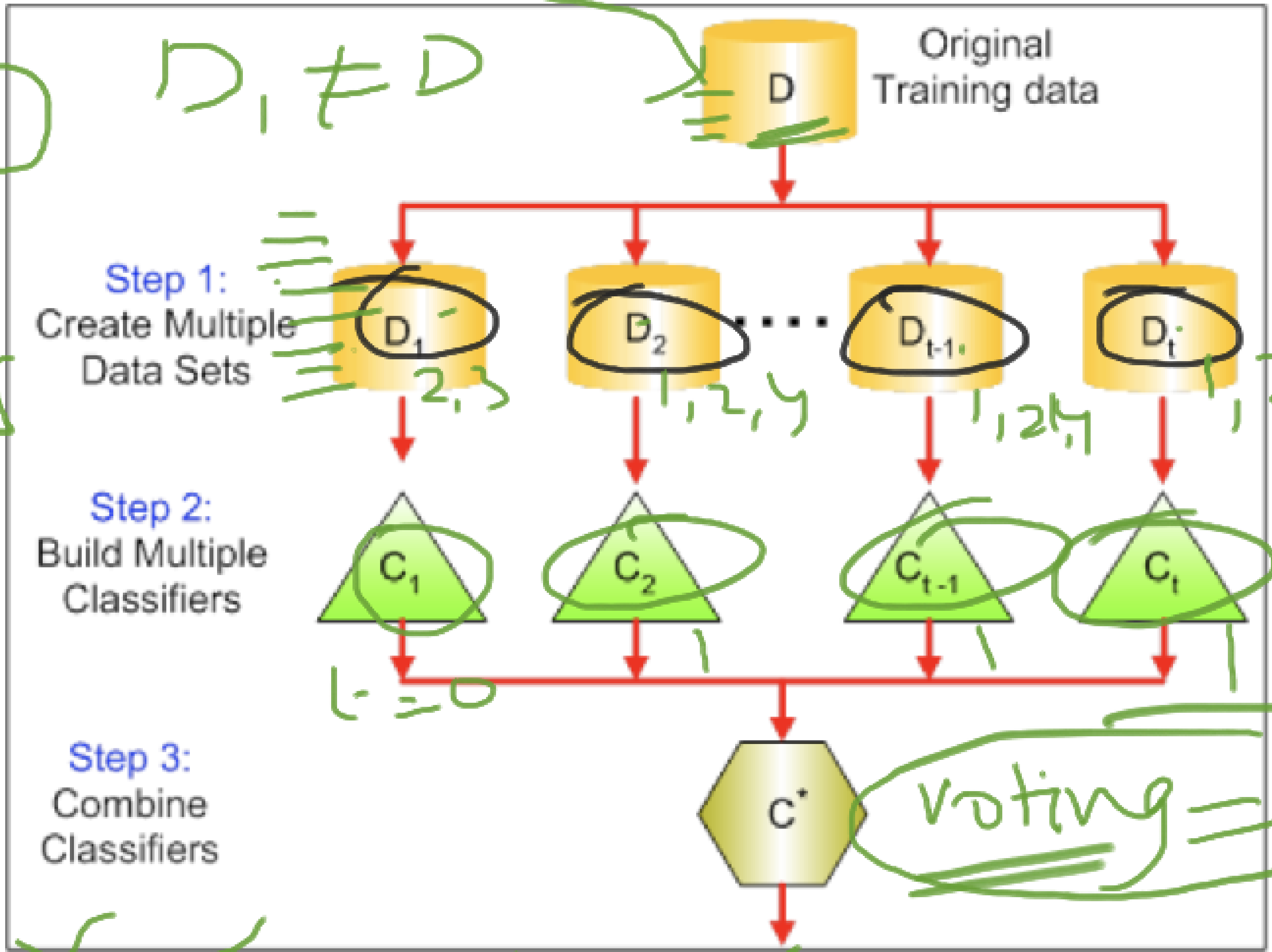
with replace

Kiy = King

$D_1 \neq D$

1000 x 10

"D"



random

1, 2, 3, 4, 5

averages

voting = 1

[C1 C2 C3 C4 C5]

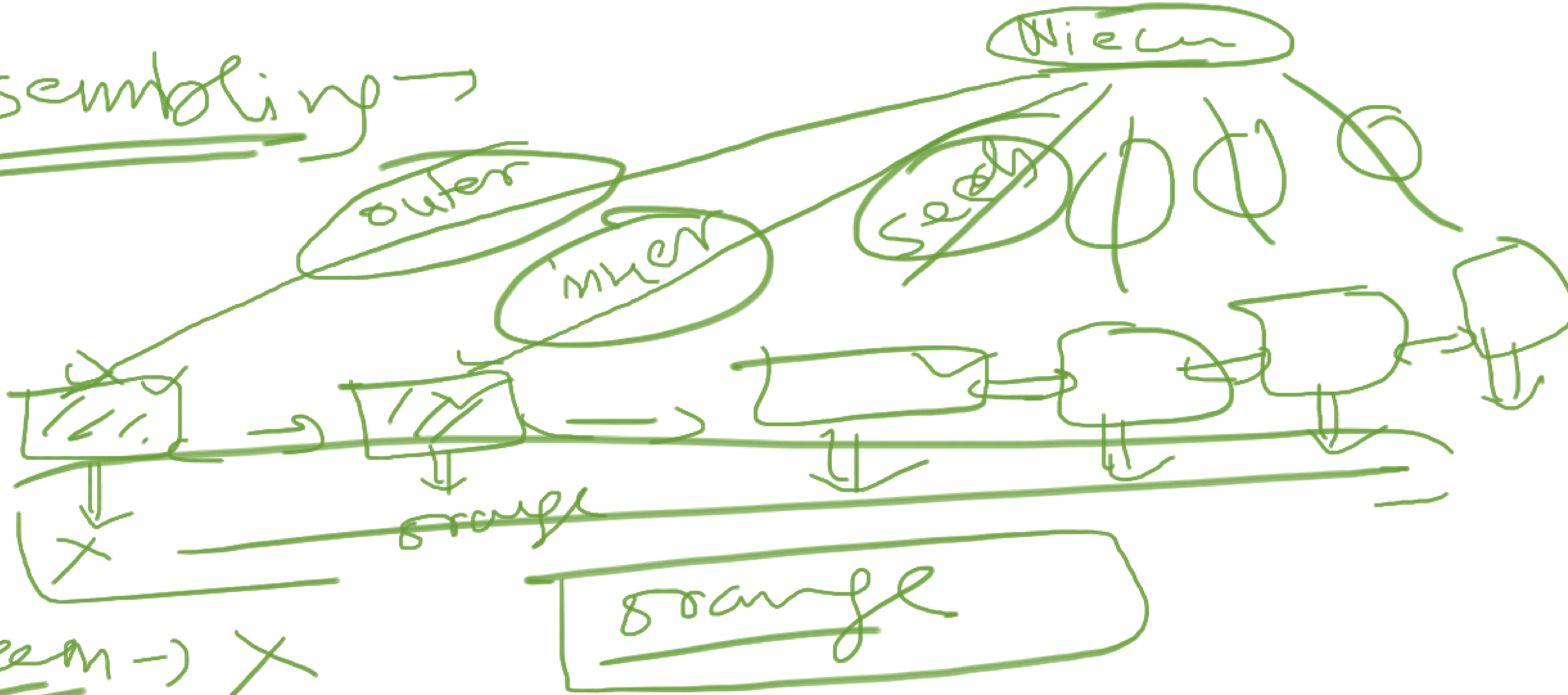
row sampling

column

Bagging → Random Forest

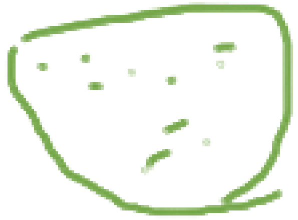
Random Row & Col

Ensembling →



green → X

Boosting $\rightarrow$ B



Niece

Utility

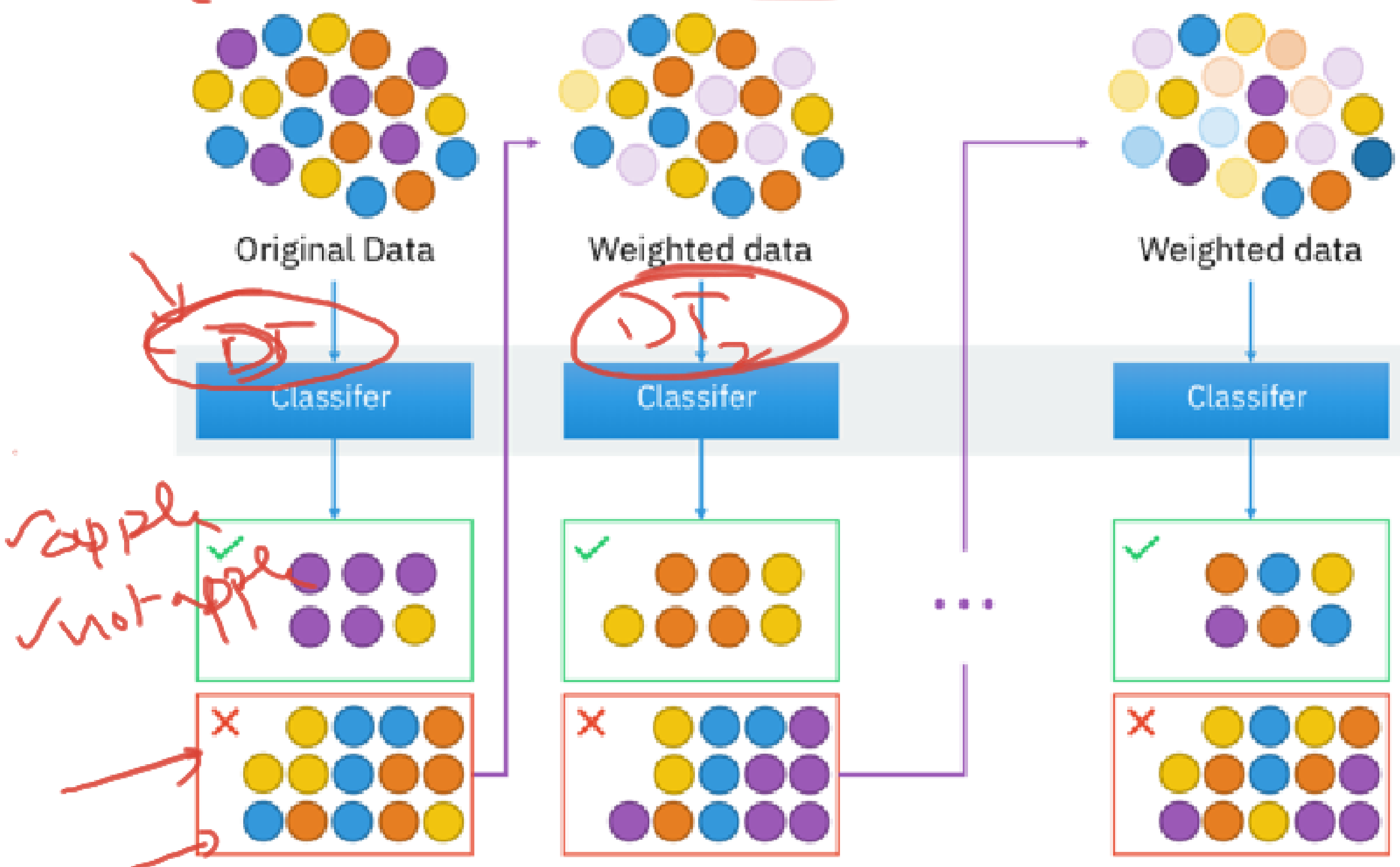
error

overlooked feature

original dataset

BOOSTING LEARNING PROCEDURE

feature



Strong Learner

Weak Learners

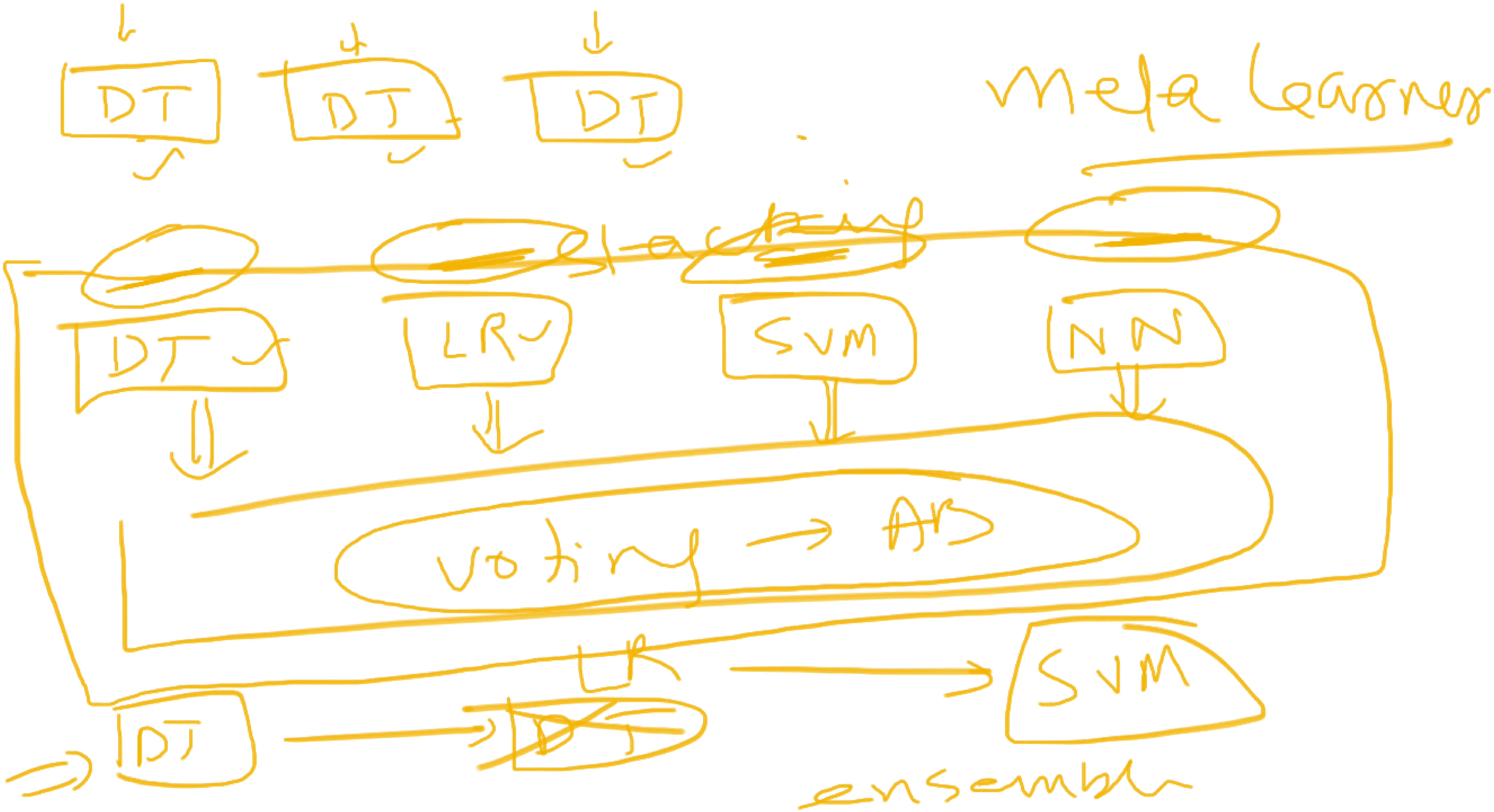
Ensemble Classifer

$$f(x) = \sum_t \alpha_t h_t(x)$$

Weight calculated by considering the last iteration's error

feature

feature → model → errors

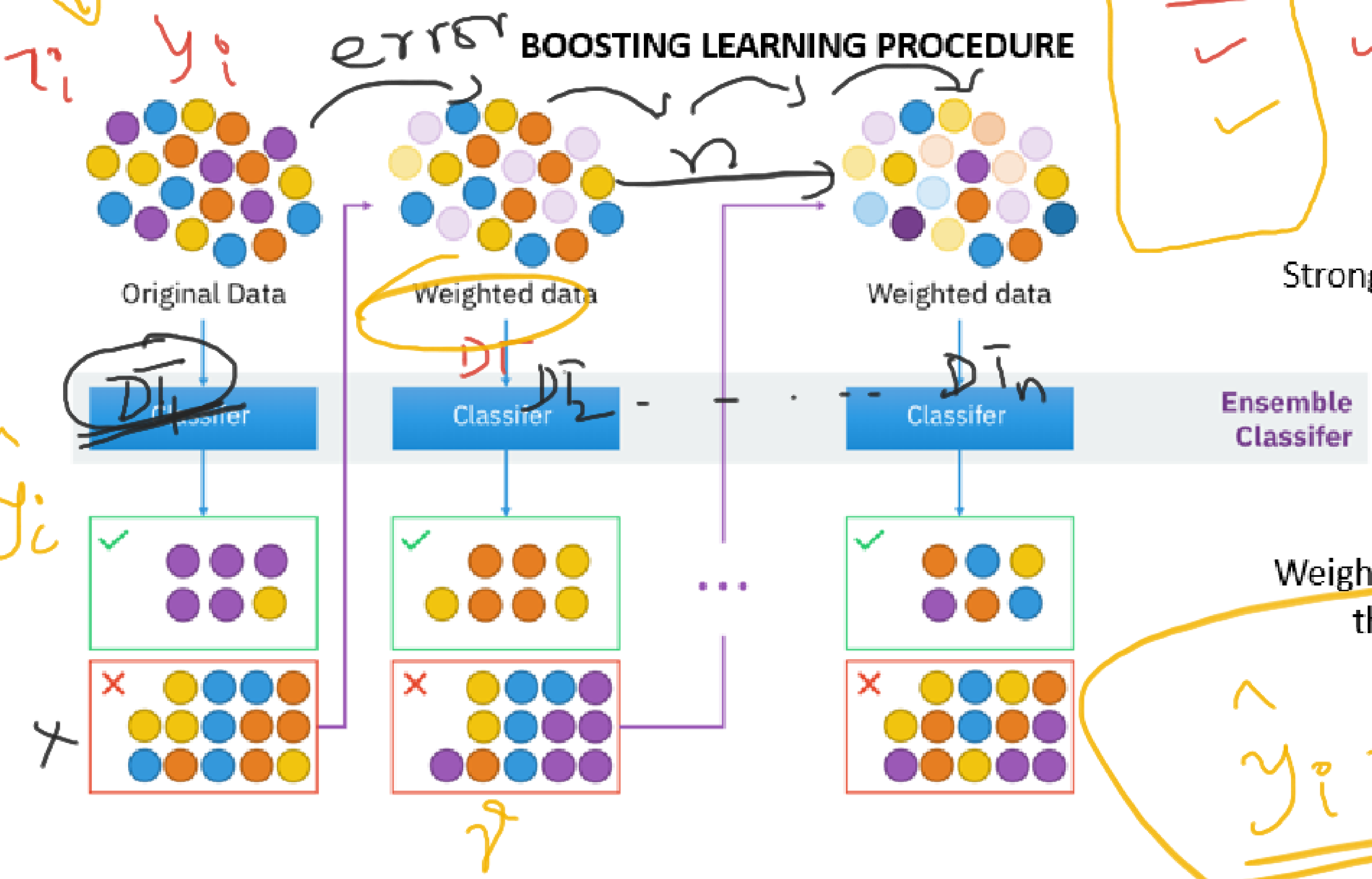

DT
DT
DT
meta learner
stacking
DT
LR
SVM
NN
voting → AB
LR
SVM
DT
~~DT~~
ensemble

~~new dataset~~

Metalearn

BOOSTING LEARNING PROCEDURE

x_i y_i $\hat{y}_i$ $(y_i - \hat{y}_i)^2$



Strong Learner

Weak Learners

$$f(x) = \sum_t \alpha_t h_t(x)$$

Weight calculated by considering the last iteration's error

$$\hat{y}_i = \hat{y}_{i1} + \gamma \hat{y}_{i2} + \gamma_2 \hat{y}_{i3}$$

Initial weights $W: W_1, W_2, \ldots, W_n = \frac{1}{n}$

pseudocode

AdaBoost

for i in $[1, M]$// M weak classifiers

fit weak classifier C^i *with sample weights* W

$$Error^i = \frac{\sum_{j=1}^{n} W_j I(C^i(X_j) \neq Y_j)}{\sum_{j=1}^{n} W_j}$$

XgBoost

f(x)

$\alpha^i = \log\left(\frac{(1-Error^i)}{Error^i}\right) + \log(K-1)$//coefficient for C^i

$W_j = W_j * e^{\alpha^i * I(C^i(X_j) \neq Y_j)}$ *for* $W_j \in W$// if wrong, increase weights

$W = W - mean(W)$// normalize weights

//output of the model is done by weighted voting, find the class with highest vote

Prediction: $\hat{Y}_j = \max_k \left(\sum_{i=1}^{i} \alpha_i I(C^i(X_j) = k)\right)$

modelling error

$W_j \rightarrow$ weights of each pt's

Wi

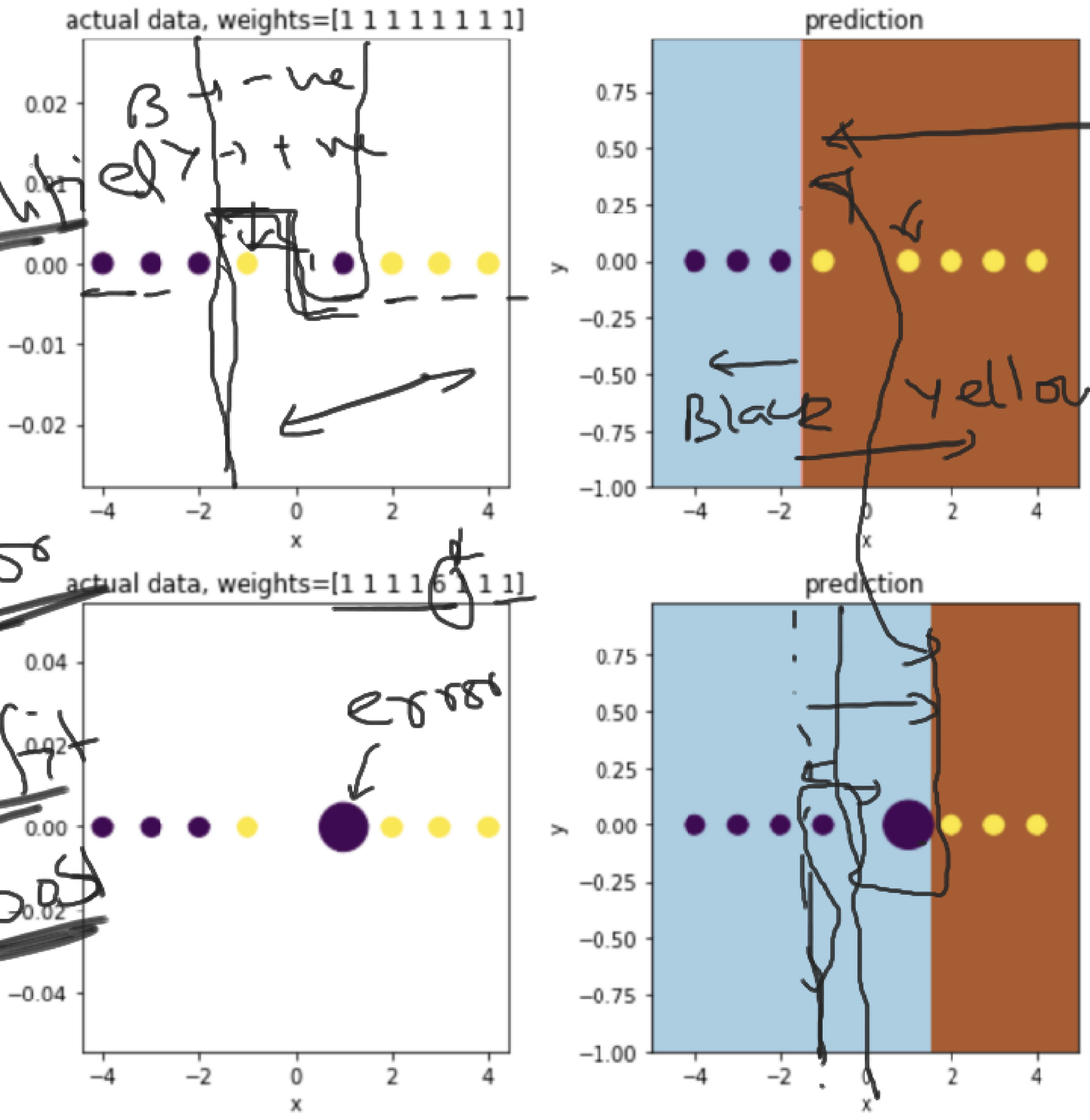
over simplified

Simp

error

overfit

XGBoost



Boundary

WT

correct

overall error

ROC-AUC   frad

Recall → +ve class (actual +ve)

$$\frac{TP}{TP+FN}$$

$$\frac{TP}{TP+FP}$$

		Actual 0	Actual 1
Pred	0	TN	FN
	1	FP	TP = 93

EM → ~~TN~~

$$\text{Recall} = \frac{\text{+ve class correctly}}{\text{Total +ve class actual}}$$

P =   T

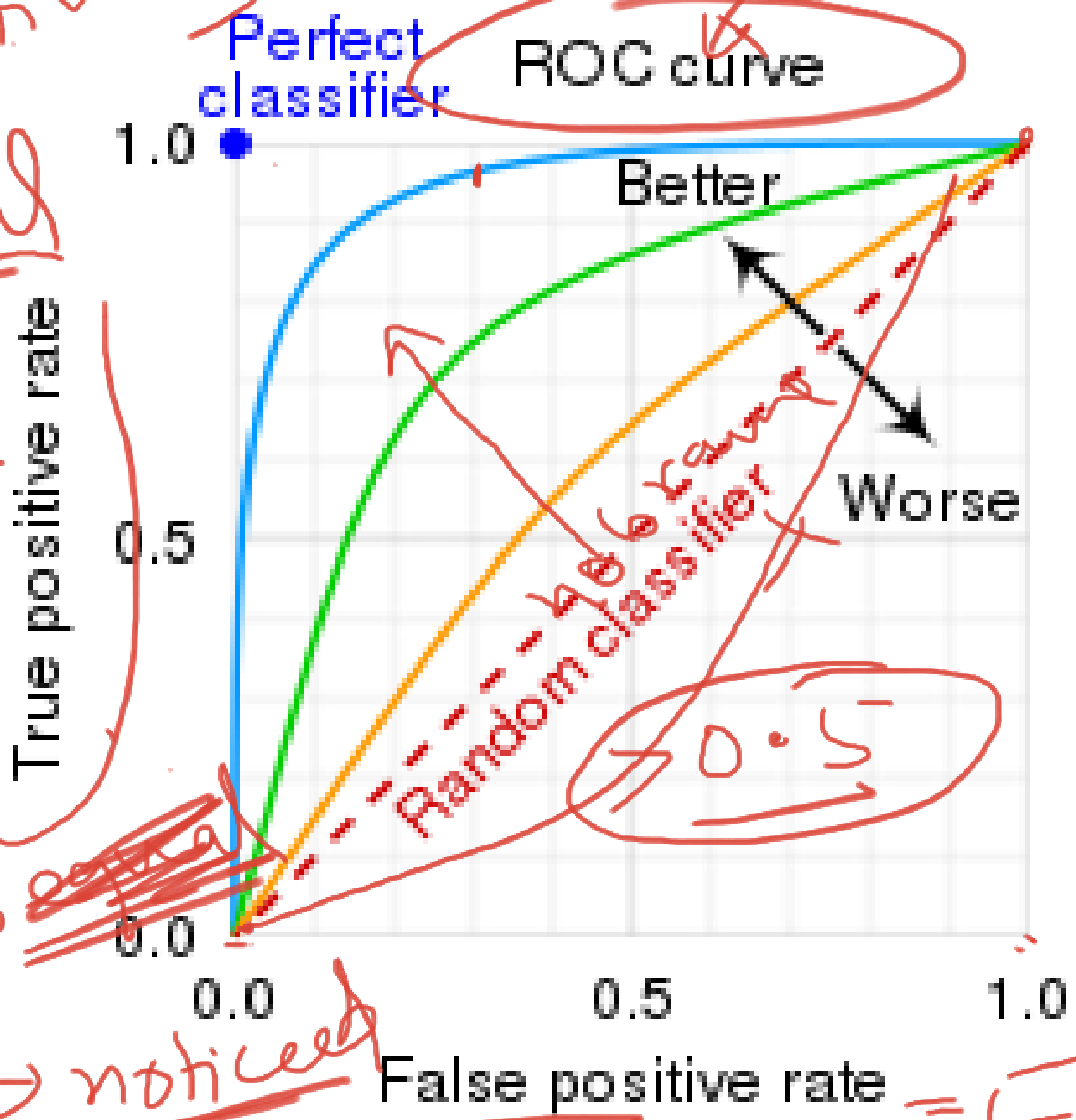
Specificity & sensitivity = Recall +ve

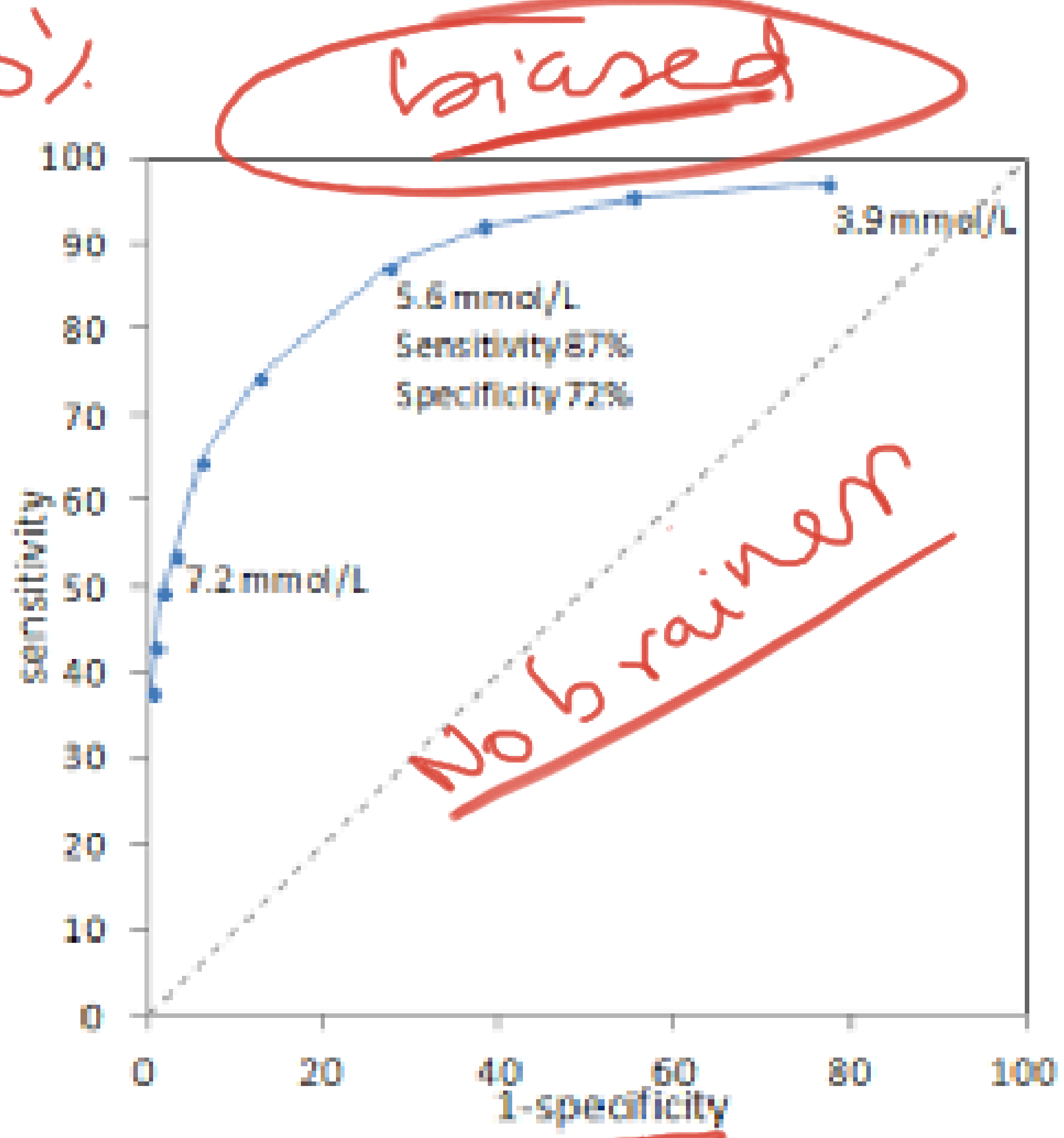
Specificity → Recall -ve class

$$\frac{TN \rightarrow \text{-ve corr}}{TN + FP \;\; \text{-ve class}}$$

sensitivity f↓ — useful 0 < ratio < 1

recall — 50%





TN
TP → noticed — equal

= 1 - specificity → recall -ve

recall -ve class

EM

ROC - AUC → threshold

P / R

adv

P > 0.5 → 1

< 0.5 → 0

ROC-AUC

↓

Biased class

0 → 10K

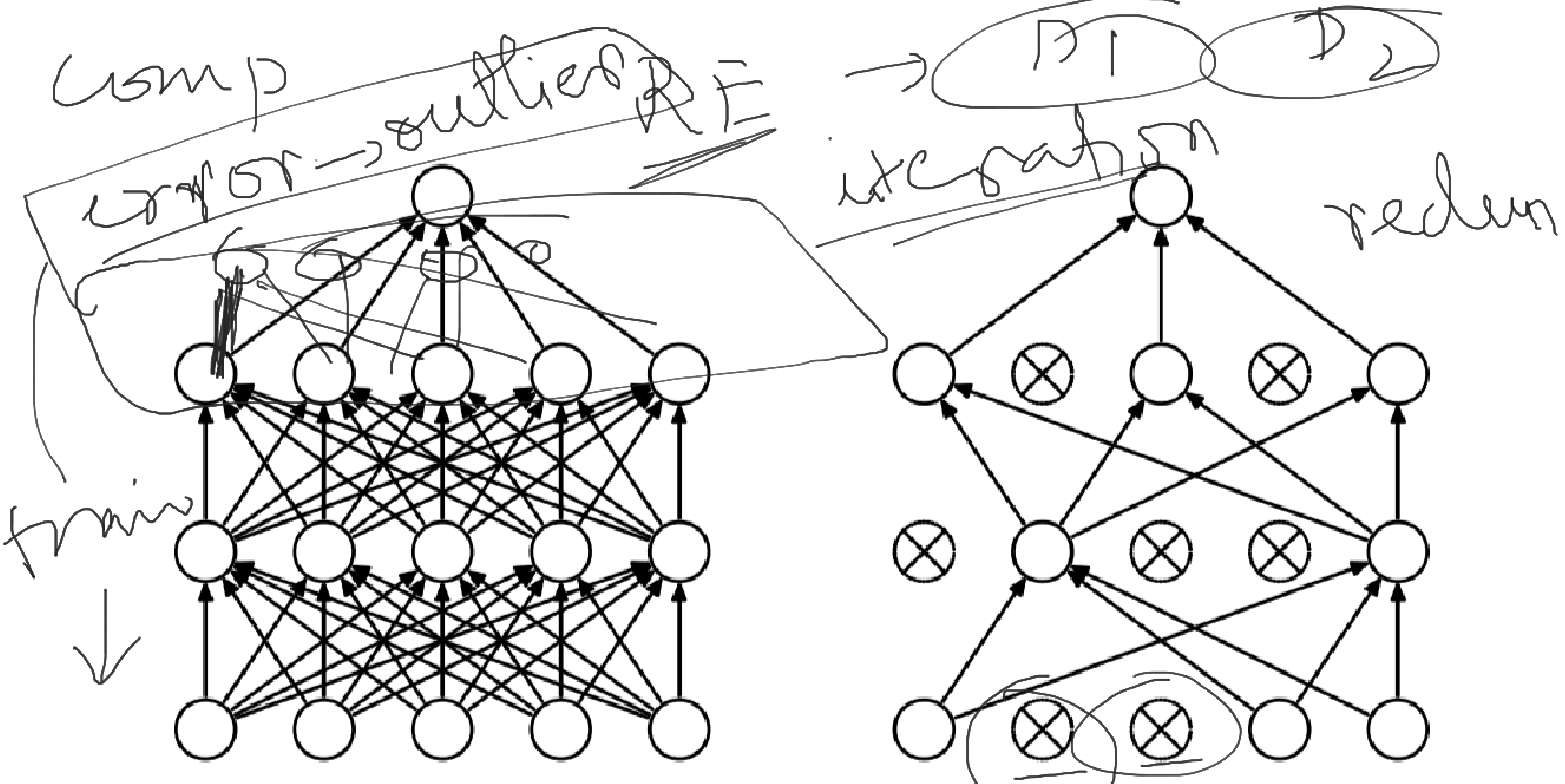
1 → 100

TP
FP
TN
FN

0.4

0.3

classific → ~~cont~~ ~~forrset~~



(a) Standard Neural Net

(b) After applying dropout.