Simple Voting Ensembles

Trained	Hard Votes	Soft Votes
C_1	0	P1 [0.3 0.6 0.1]
C_2	0	P2 [0.4 0.4 0.2]
C_3	1	P3 [0.4 0.4 0.2]
C_4	2	P4 [0.2 0.6 0.2]

$X →$

$$[0.2 \quad 0.6 \quad 0.2]$$
$$q_1 \qquad q_2 \qquad q_3$$

majority
(to draw decision)
↓
like election
↓
issue with ties

→ probability distribution
divide rote in weights

q_2 is highest]
fractional votes
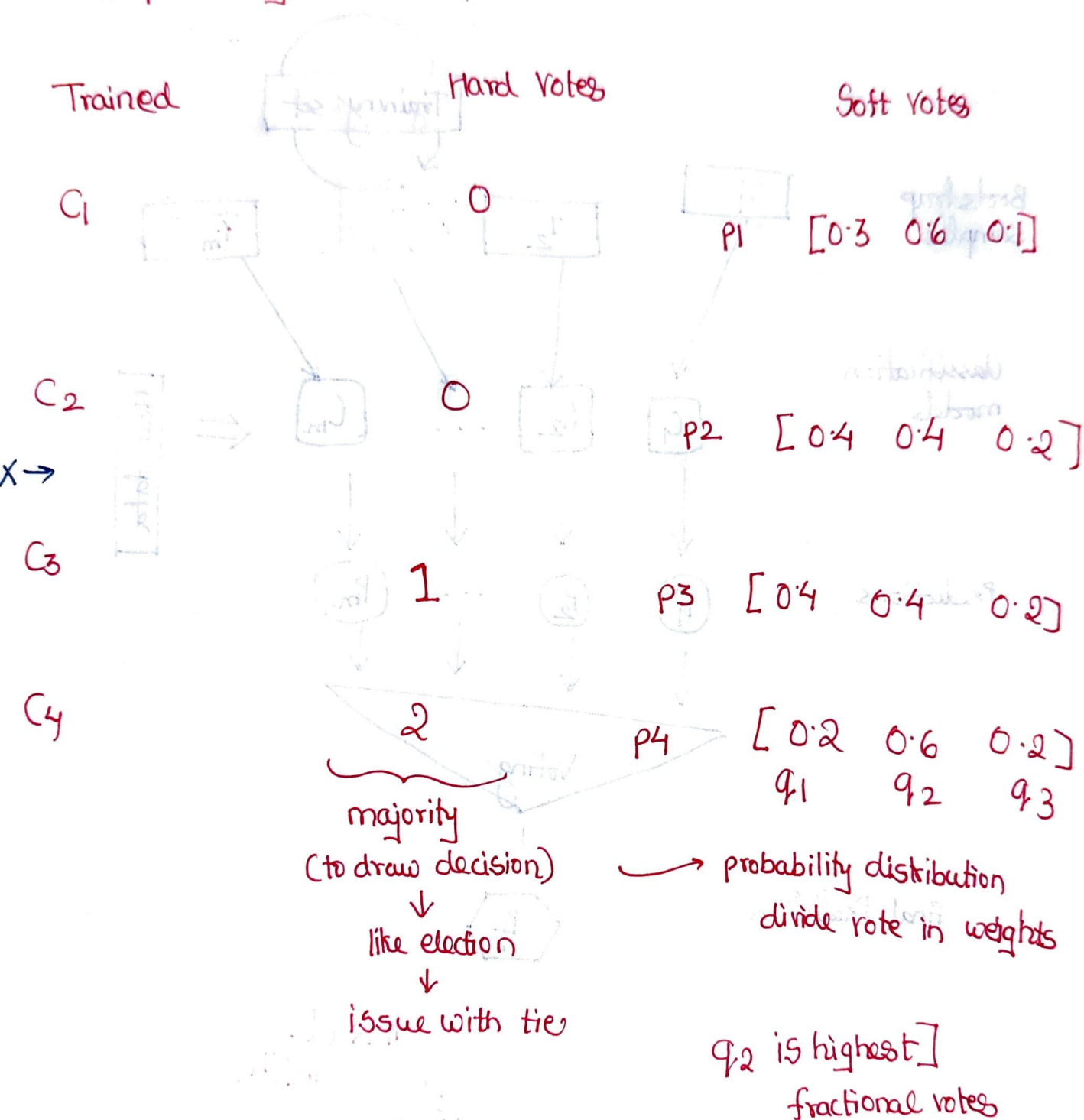
→ we can also take geometric mean.

Bootstrapping aggregation (bagging)

Training set

Bootstrap samples

T_1 T_2 . . . . T_m

Classification models

C_1 C_2 . . . C_m ⟸ New data

Predictions

P_1 P_2 . . . P_m

Voting

Final Prediction

P_f

$r \times d$.

with.

↙ without
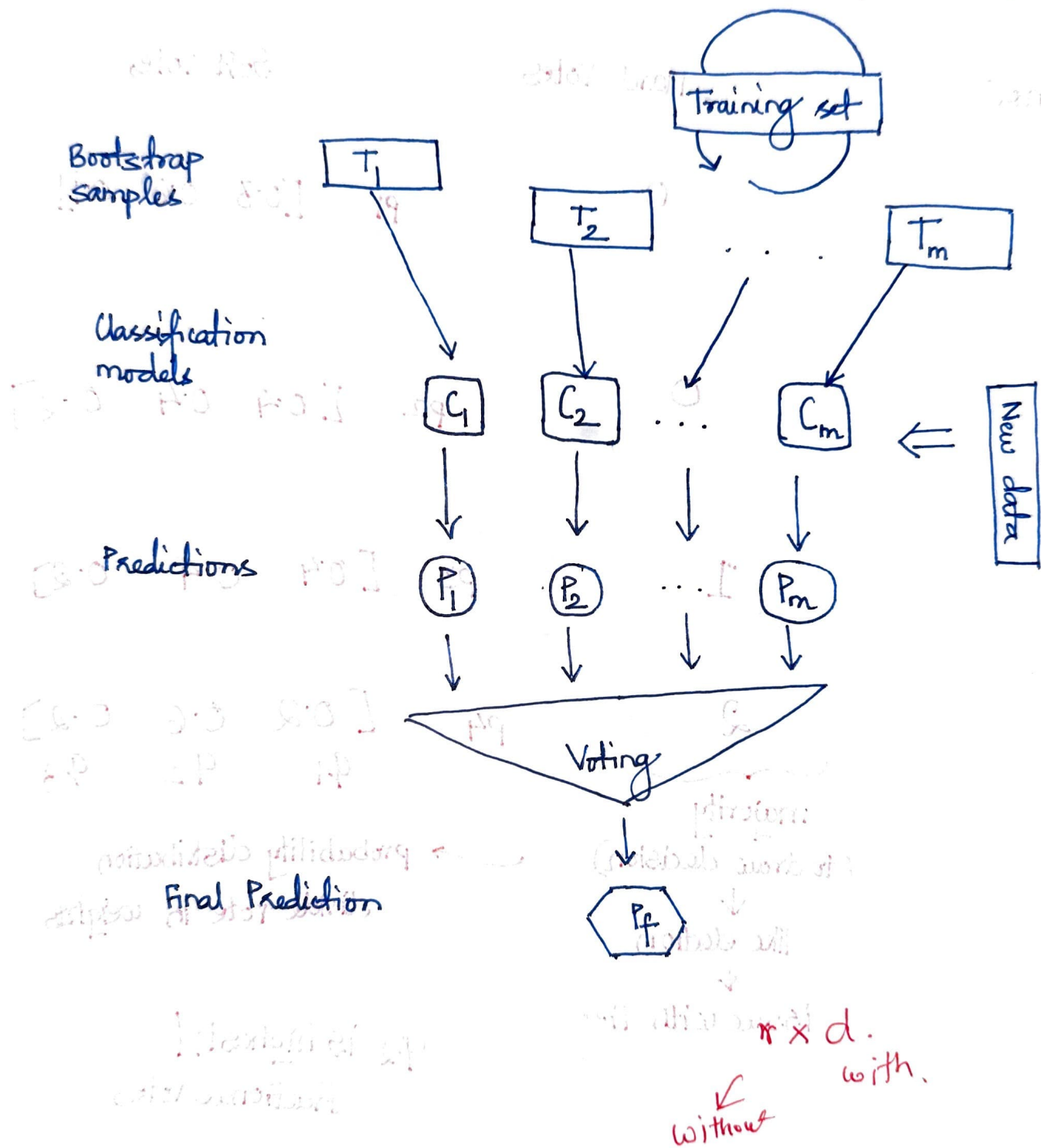
Adaptive Boost (AdaBoost)

$$1 \quad C_1$$
$$0.8 \quad C_2$$
$$0.6 \quad C_3$$
$$0 \quad \vdots$$
$$\vdots$$
$$0.1 \quad C_n$$
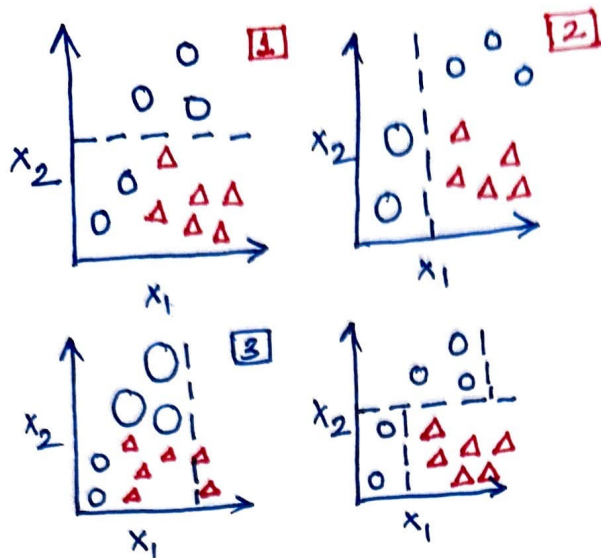
Adaboost Pseudo Code

- Initially, all n points are given equal weights, $w_i^{(0)}$, for $i = 1, \ldots n$
- For $t = 1, \ldots, m$
- Train base classifier b_t on X with importance weights $w^{(t-1)}$
- Let ϵ_t be the training error of b_t
- Calculate an importance weight for classifier t: $a_t = \frac{1}{2} \ln \frac{1 - \epsilon_t}{\epsilon_t}$
- Update the weight of point i: $w_i^{(t)} = w_i^{(t-1)} e^{-a_t y_i b_t (x_i)}$
- Normalize the weights, i.e. let $S = \sum_{i=1}^{n} w_i$, and $w_i = w_i / S$