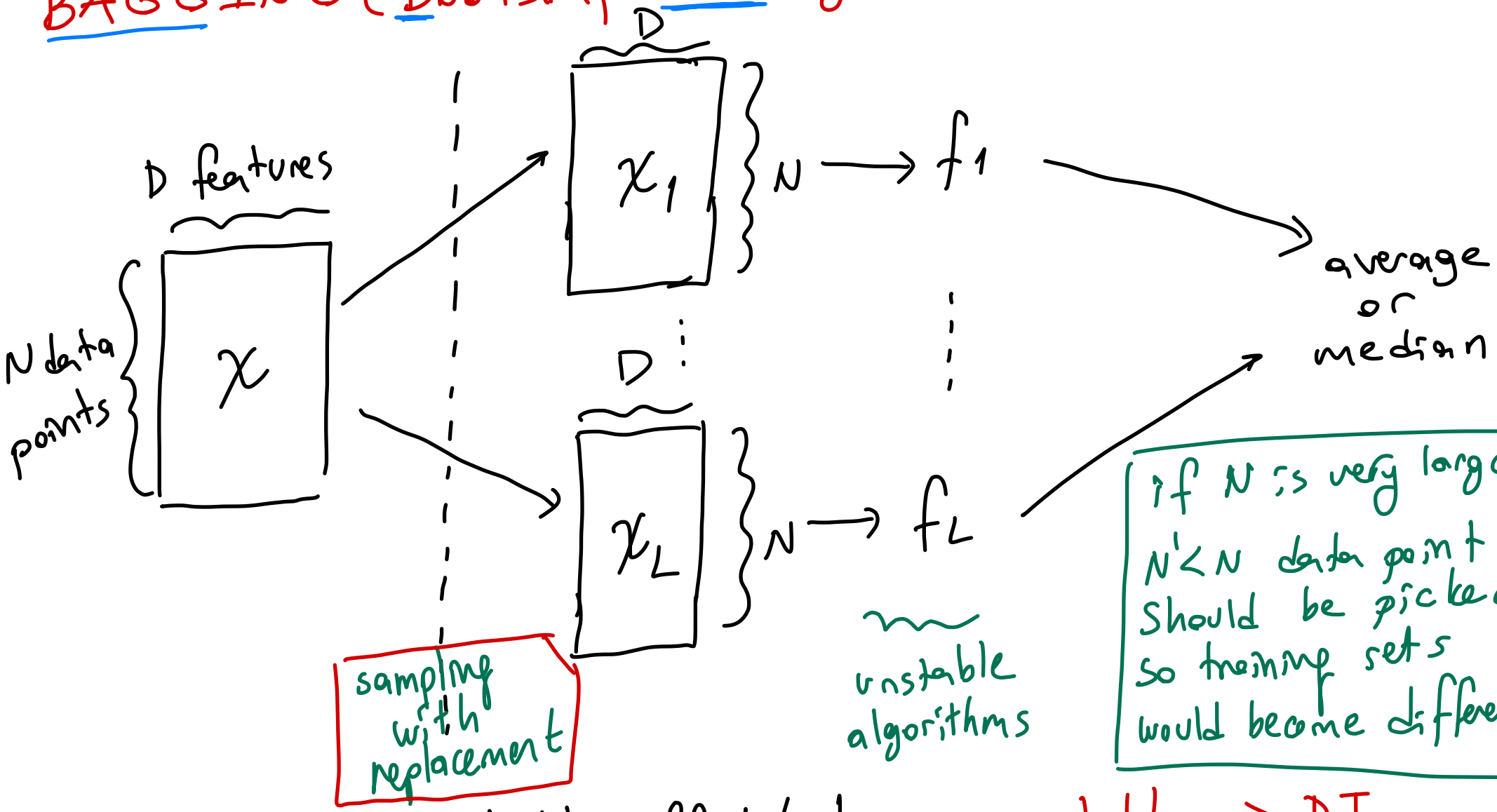
BAGGING (Bootstrap AGGregation)

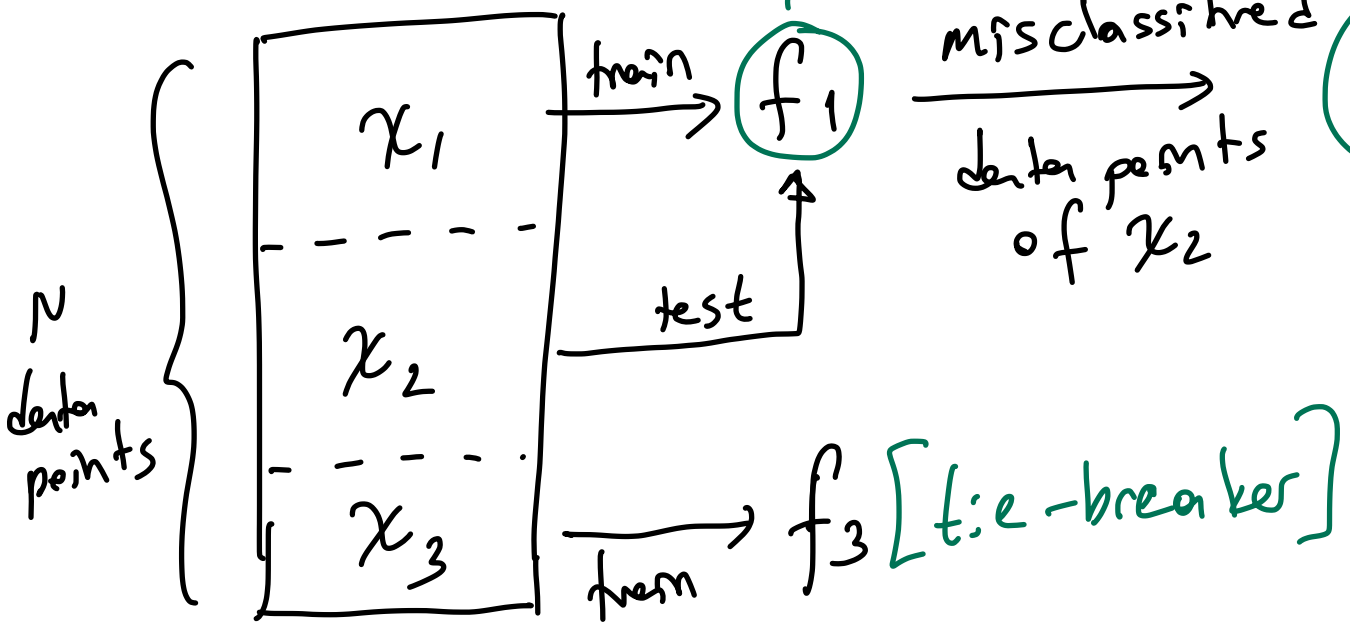


Unstable Algorithm: highly affected by small changes in the training data set.

unstable ⇒ DT

stable ⇒ k-NN

BOOSTING:



X_{N+1} (test data point)

$f_1(X_{N+1})$ $f_2(X_{N+1})$

(1) If they agree on their decisions, no problem and use the predicted label.

(2) If they do not agree, use $f_3(X_{N+1})$ as the predicted label.

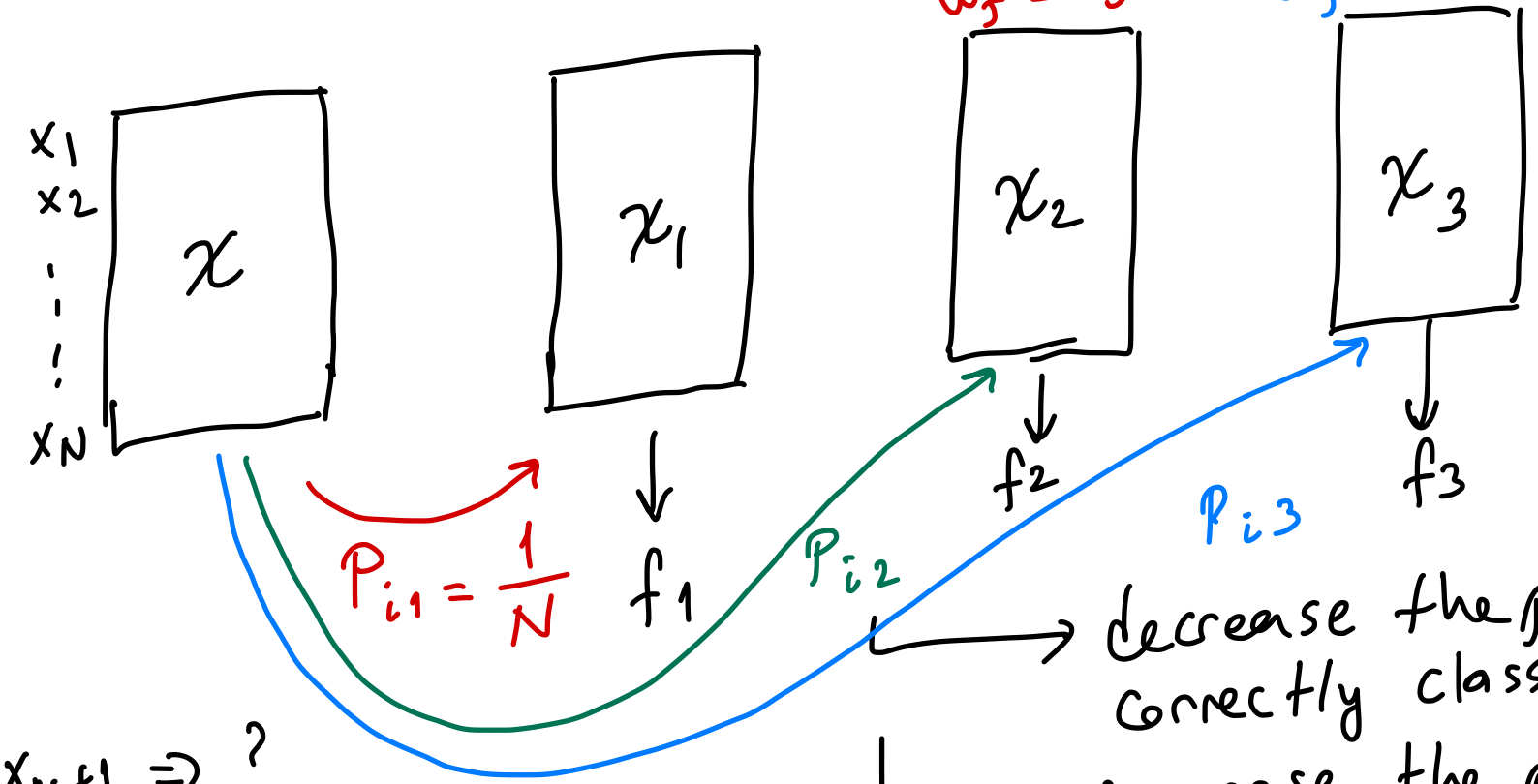
Ada Boost: modify the probabilities of drawing instances as a function of the error.

P_{ij} = the probability that the instance x_i is selected by classifier f_j.

$\epsilon_j = 0.20$
$\beta_j = 0.20/0.80$
$w_j = \log(4)$

$\epsilon_j = 0.50$
$\beta_j = 0.50/0.50$
$w_j = \log(1) = 0$

$\epsilon_j = 0.01$
$\beta_j = 0.01/0.99$
$w_j = \log[99]$



$w_j = \log[1/\beta_j]$

$\beta_j = \frac{\epsilon_j}{1-\epsilon_j}$

ϵ_j = error rate.

P_{i3}

→ decrease the probabilities for correctly classified data points

→ increase the probabilities for incorrectly classified data points

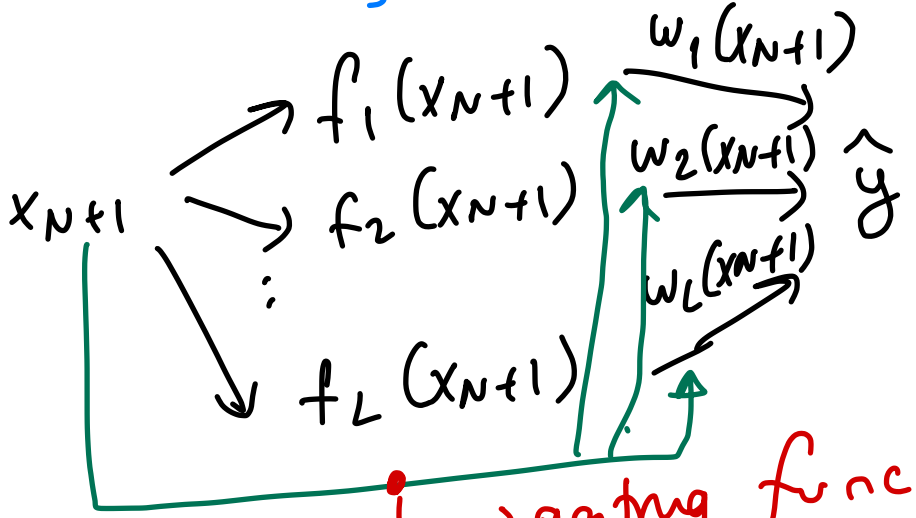
$x_{N+1} \Rightarrow ?$

$$f(x_{N+1}) = w_1 f_1(x_{N+1}) + w_2 f_2(x_{N+1}) + \cdots + w_L f_L(x_{N+1})$$

Mixture of Experts (MoE):

Voting $\Rightarrow \hat{y} = \sum_{j=1}^{L} w_j f_j(x_{N+1})$ — constant over the input space.

MoE $\Rightarrow \hat{y} = \sum_{j=1}^{L} w_j(x_{N+1}) \cdot f_j(x_{N+1})$



w_j's will be assigned by the gating function.

Competitive:

→ $w_1, w_2, \dots, w_L$ are usually sparse (i.e., mostly zero)

→ one or some of them are nonzero.

Softmax: $w_j = \dfrac{\exp(v_j^T \cdot x + v_{j0})}{\sum_{k=1}^{L} \exp(v_k^T \cdot x + v_{k0})}$

Cooperative:

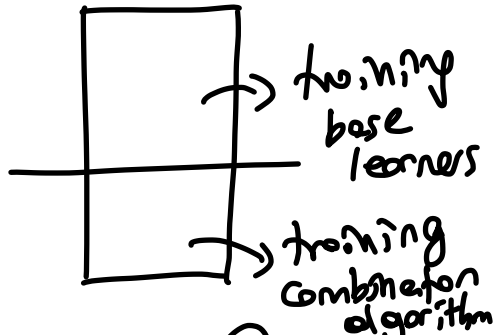
→ $w_1, w_2, \dots, w_L$ are assumed to be independent.

Sigmoid: $w_j = \dfrac{1}{1 + \exp\left[-(v_j^T \cdot x + v_{j0})\right]}$

Stacked Generalization

Voting $\Rightarrow \hat{y} = \sum_{j=1}^{L} w_j f_j(x_{N+1})$

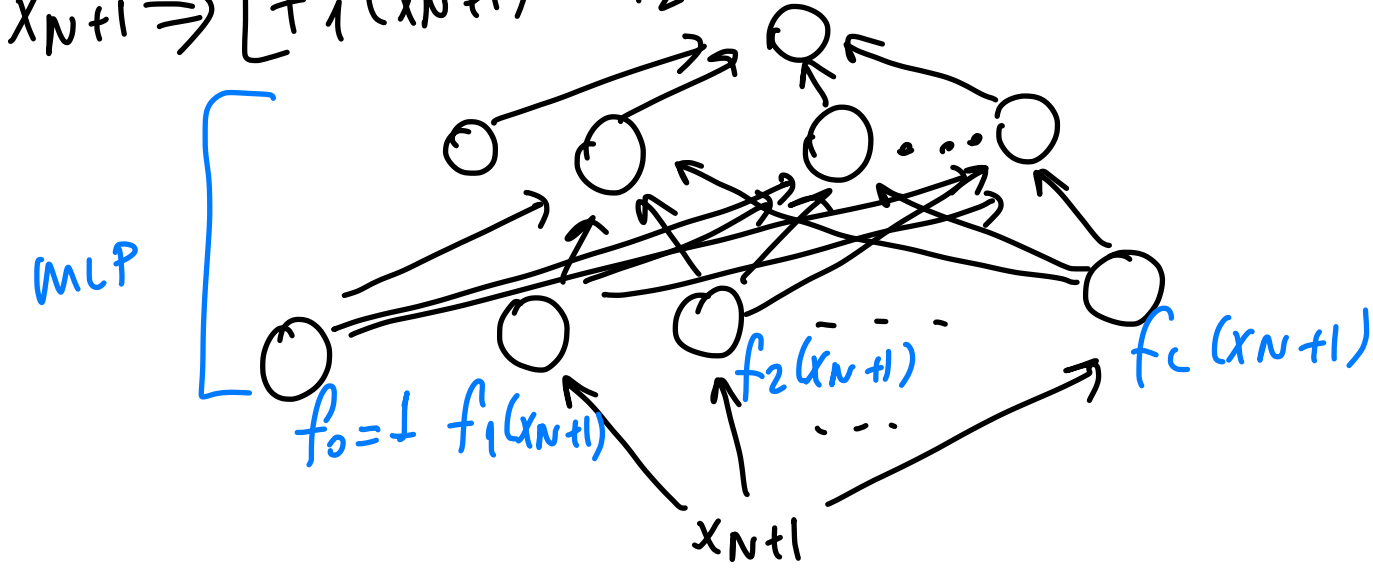
MoE $\Rightarrow \hat{y} = \sum_{j=1}^{L} w_j(x_{N+1}) \cdot f_j(x_{N+1})$

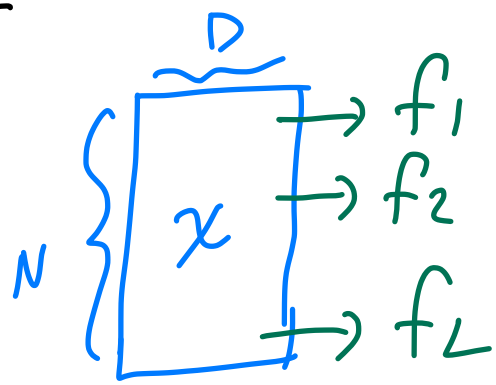


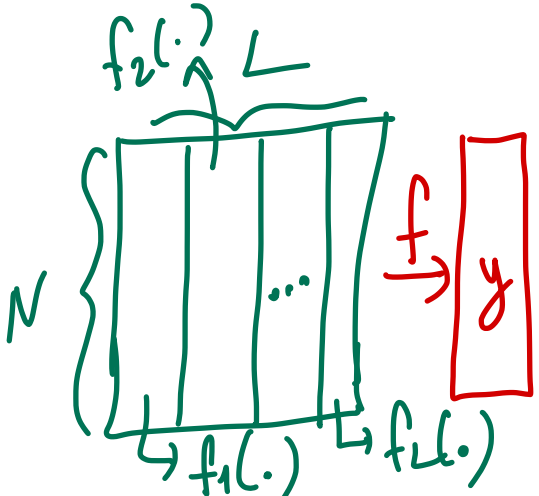
Stacked Generalization $\Rightarrow \hat{y} = \boxed{f}\left(f_1(x_{N+1}), f_2(x_{N+1}), \dots, f_L(x_{N+1})\right)$

$\boxed{f}$ → nonlinear algorithm

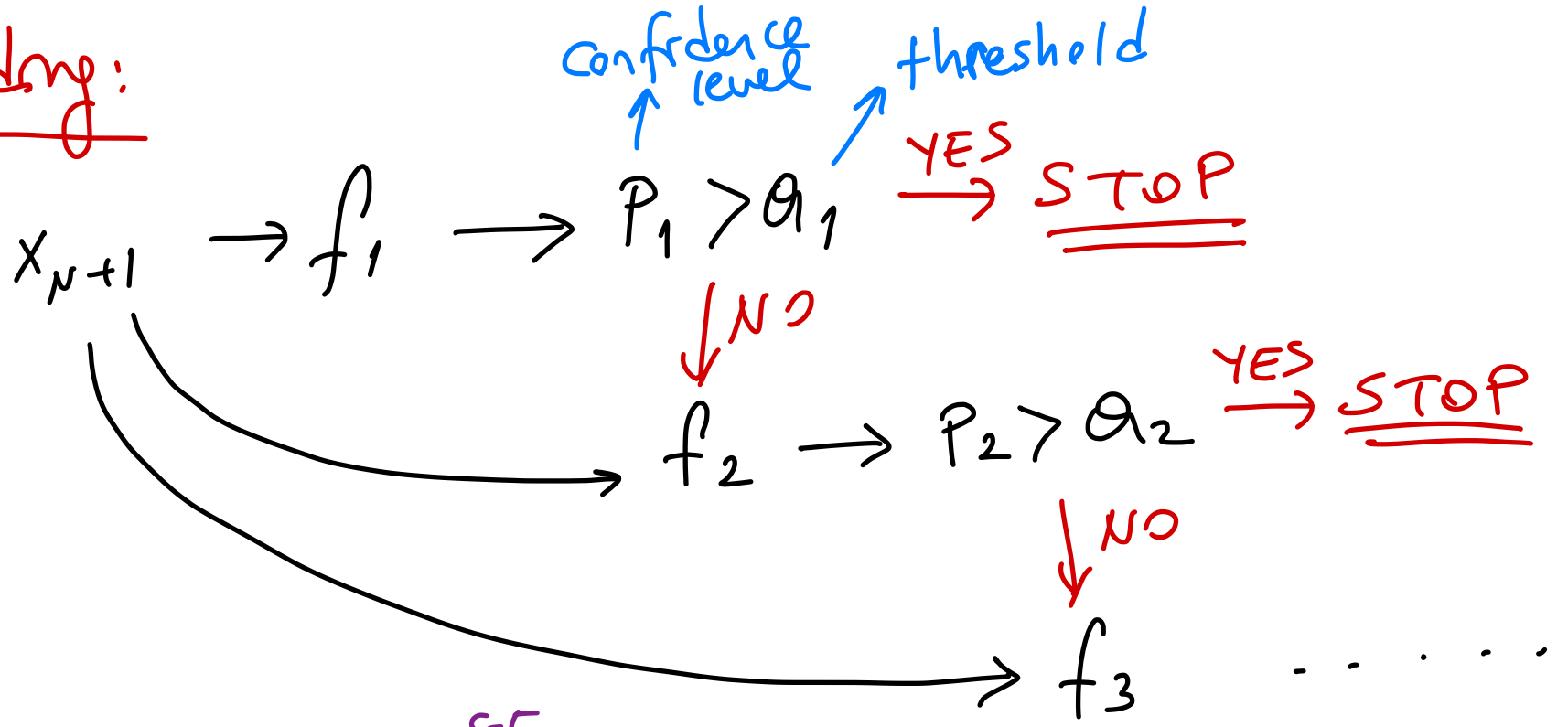
$x_{N+1} \Rightarrow \left[f_1(x_{N+1}) \quad f_2(x_{N+1}) \dots f_L(x_{N+1})\right]^T$







Cascading:



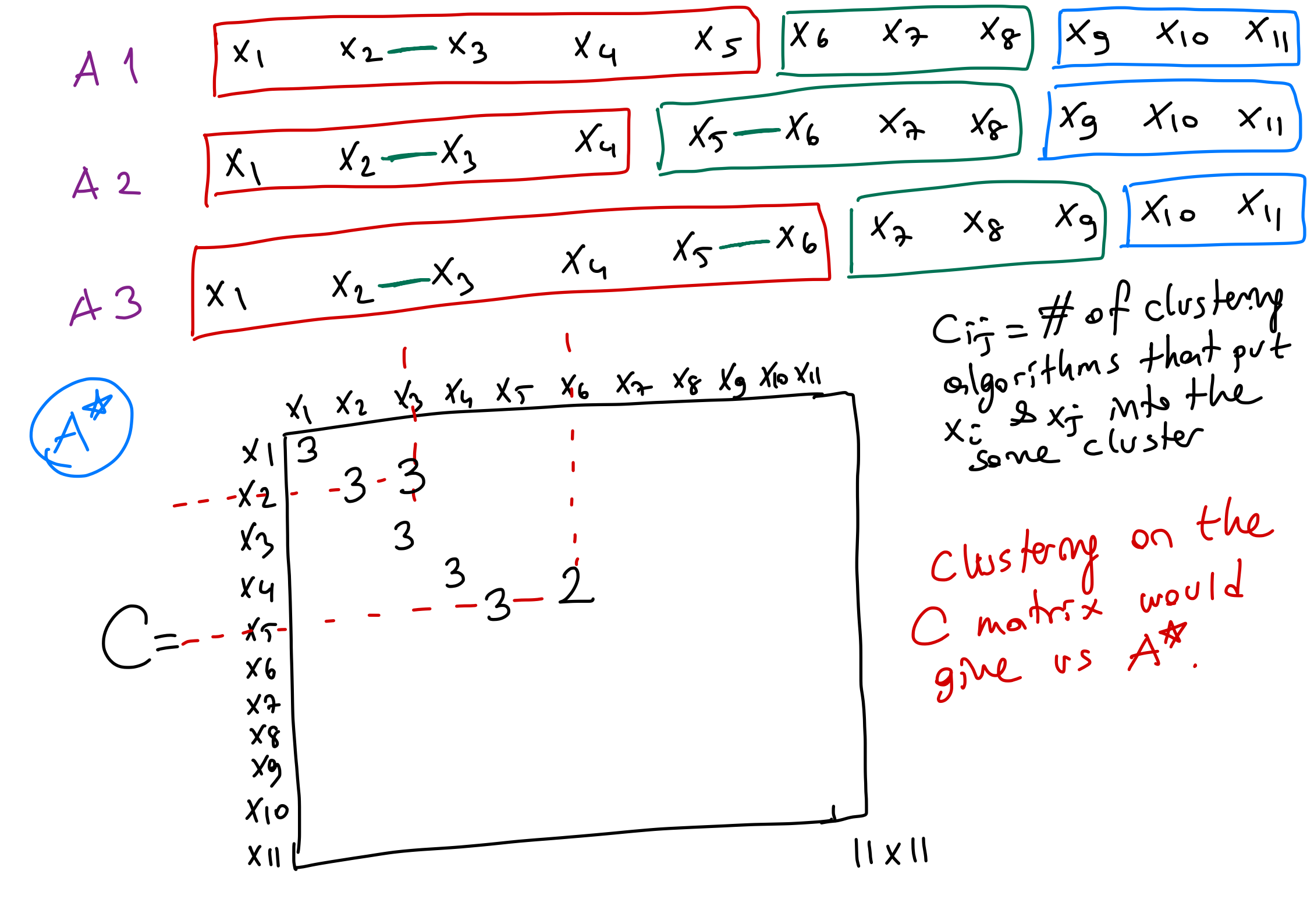
0.95 0.90 0.85

$\theta_1 > \theta_2 > \theta_3 > - - - - - -$

decreasing thresholds

$x_{N+1} \xrightarrow{f_1} P_1 = 0.98 \Rightarrow$ we are confident enough, let's STOP.
since $0.98 > 0.95$

$x_{N+1} \xrightarrow{f_1} P_1 = 0.92 \xrightarrow{f_2} P_2 = 0.91 \Rightarrow$ we are confident enough, let's STOP.
since $0.92 < 0.95$ since $0.91 > 0.90$


A 1
x1 x2 x3 x4 x5
x6 x7 x8
x9 x10 x11
A 2
x1 x2 x3 x4
x5 x6 x7 x8
x9 x10 x11
A 3
x1 x2 x3 x4 x5 x6
x2 x8 x9
x10 x11
A*
C =
x1 x2 x3 x4 x5 x6 x7 x8 x9 x10 x11
x1 3
x2 3 3
x3 3
x4 3 2
x5 3
x6
x7
x8
x9
x10
x11
11 x 11
Cij = # of clustering algorithms that put xi & xj into the same cluster
Clustering on the C matrix would give us A*.