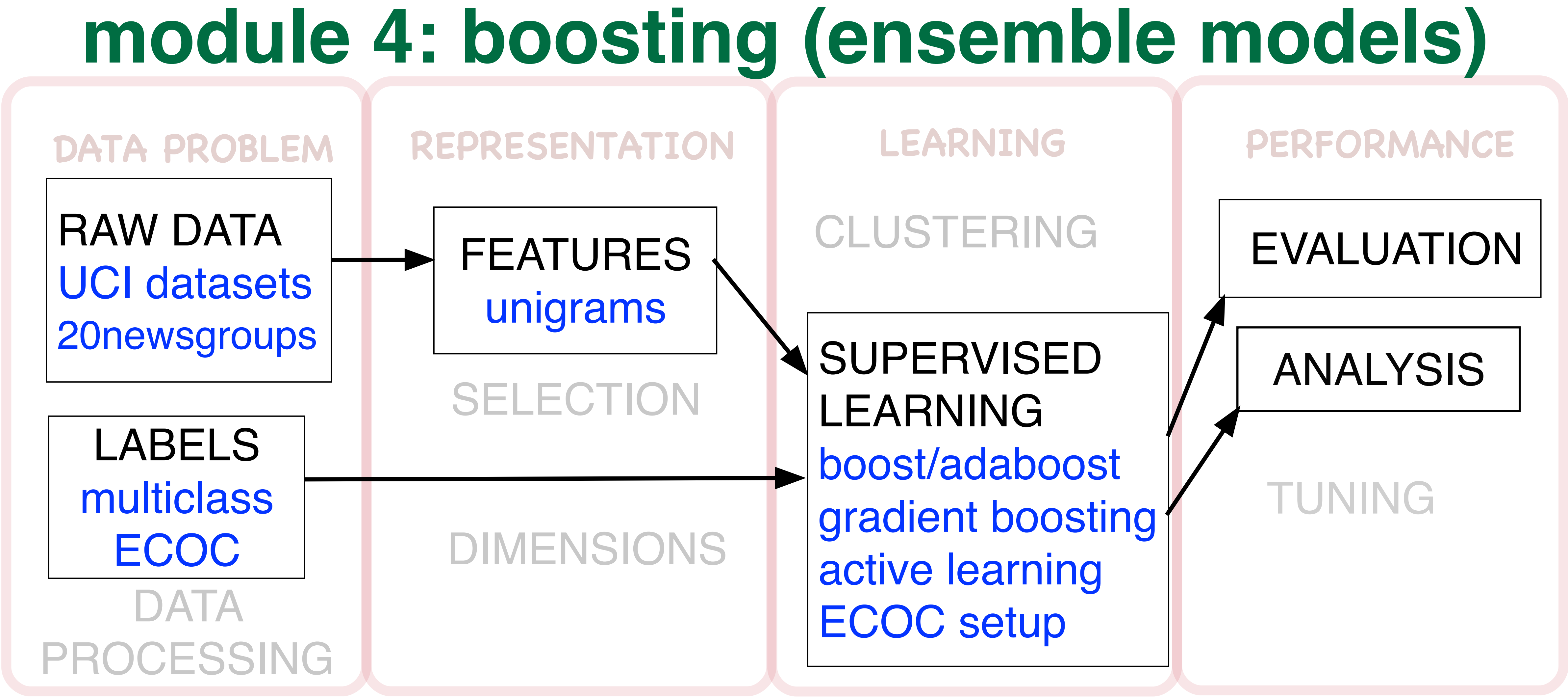
Boosting (ensemble)

Module 4 – Ensemble classifiers – Objectives



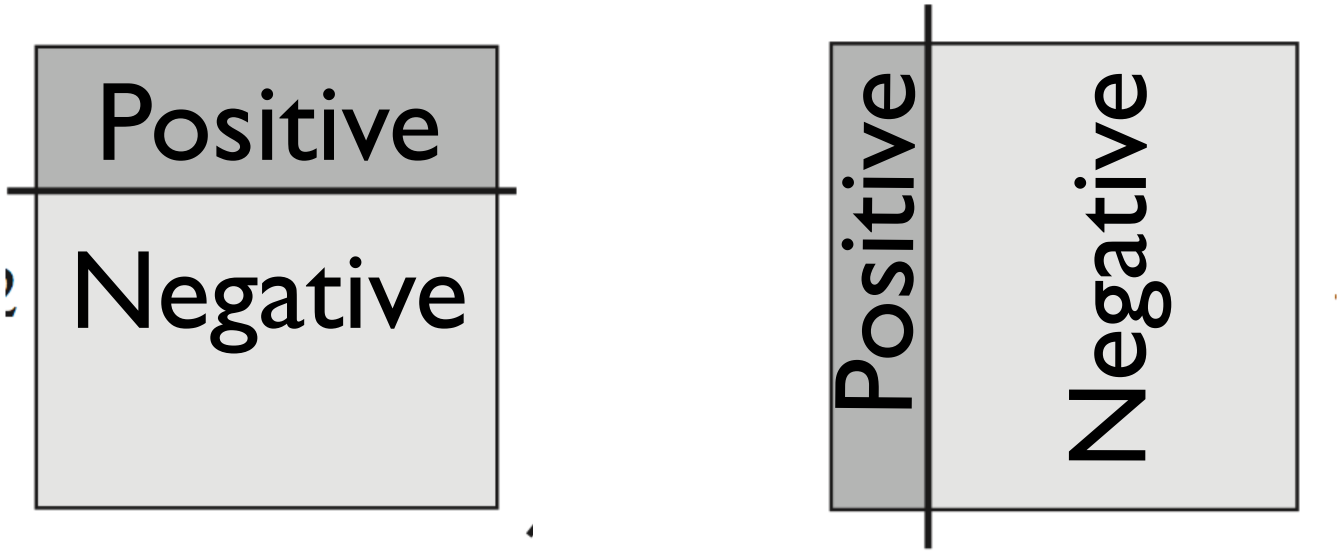
- BOOSTING: combine weak/simple classifiers into a powerful one
- Bagging: combine classifiers by sampling training set
- Active Learning: select the datapoints to train on
- ECOC for Multiclass data : introducing the 20Newsgroups dataset of articles
- VC dimension as a measure of classifier complexity

Weak Learners

- Need not to be very accurate
- Better than random guess
- Examples:
  - Decision trees/Decision stump
  - Neural Network
  - Logistic regression
  - SVM
  - Essentially any classifier

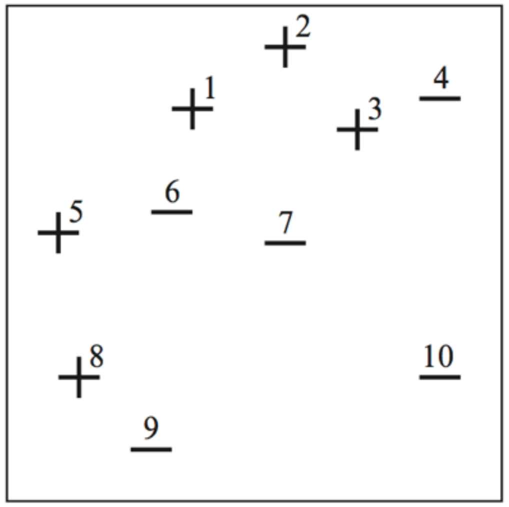
Decision Stump

- 1-Level decision tree
- A simple test based on one feature
- Eg: If an email contains the word "money", it is a spam; otherwise, it is a non-spam
- moderately accurate
- Geometry: horizontal or vertical lines



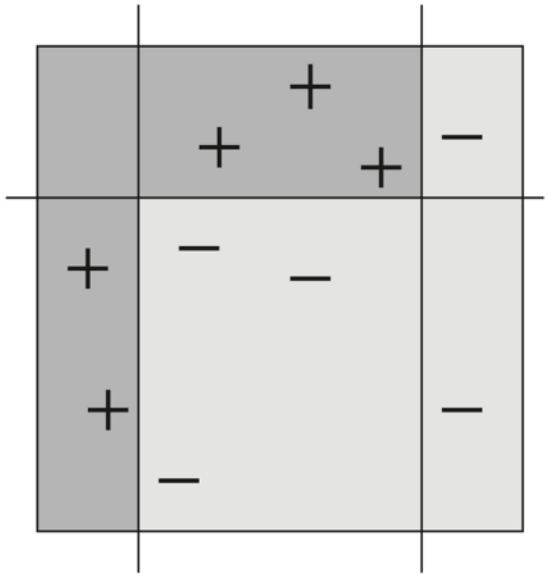
Limitation of Weak Learner

- Might not be able to fit the training data well (high bias)
- Example: no single decision stump can classifier all the data points correctly



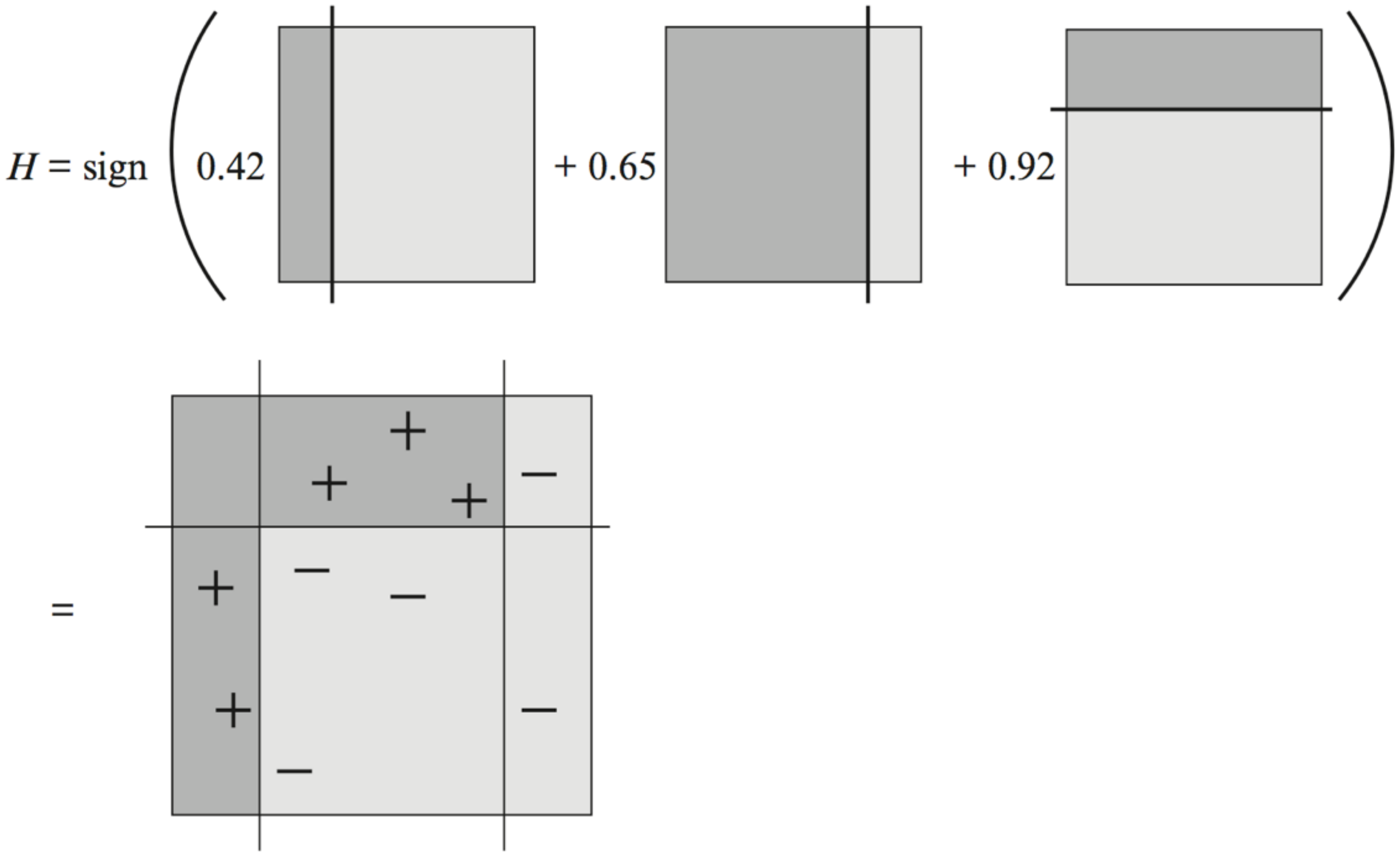
Can weak learners combine to do better?

- Can we separate the positive data from the negative data by drawing several lines?
- Yes, we can!



Can weak learners combine to do better?

- It turns out this complicated classifier can be expressed as a linear combination of several decision stumps



An analogy of Committee

- A weak Learner = a committee member
- Combination of weak leaners(ensemble) = a committee
- A Weak learner's decision hypothesis = a committee member's judgement
- Ensemble's decision hypothesis = a committee's decision
- A combination of weak learners often classifies better than a single weak learner = a committee often makes better decisions than a single committee member

Idea: Generating diverse weak leaners

- adaBoost picks its weak learners h in such a fashion that each newly added weak learner is able to infer something new about the data

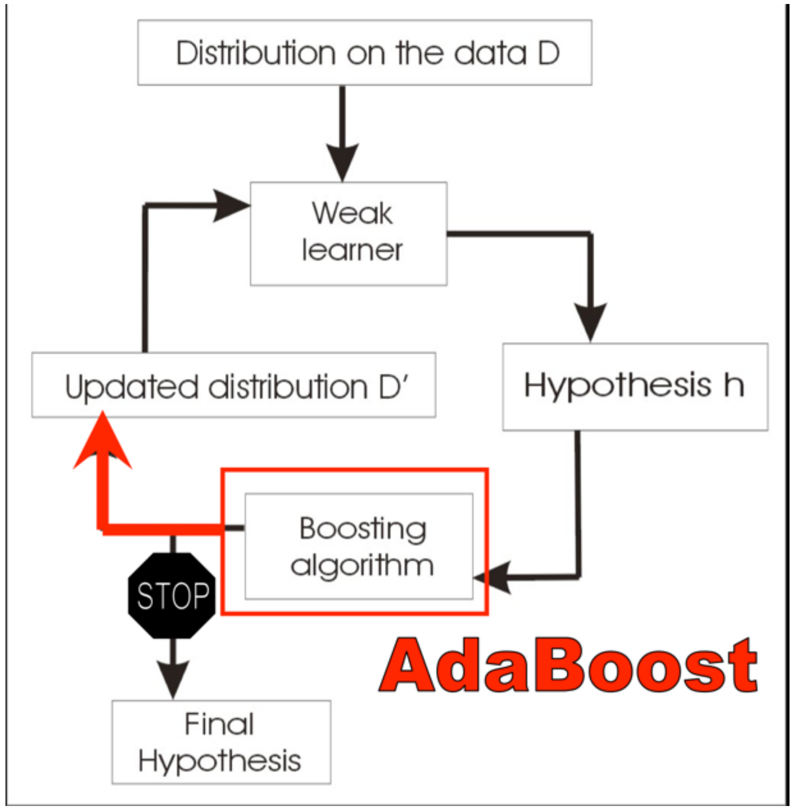
- adaBoost maintains a weight distribution D among all data points. Each data point is assigned a weight D(i) indicting its importance

- by manipulating the weight distribution, we can guide the weak learner to pay attention to different part of the data

Idea: Generating diverse weak leaners

- AdaBoost proceeds by rounds
- in each round, we ask the weak learner to focus on hard data points that previous weak learners cannot handle well

- Technically, in each round, we increase the weights of misclassified data points, and decrease the weights of correctly classified data points



- AdaBoost init: uniform weight distribution D on datapoints
- AdaBoost loop:
  - train weak learner h according to current weights D

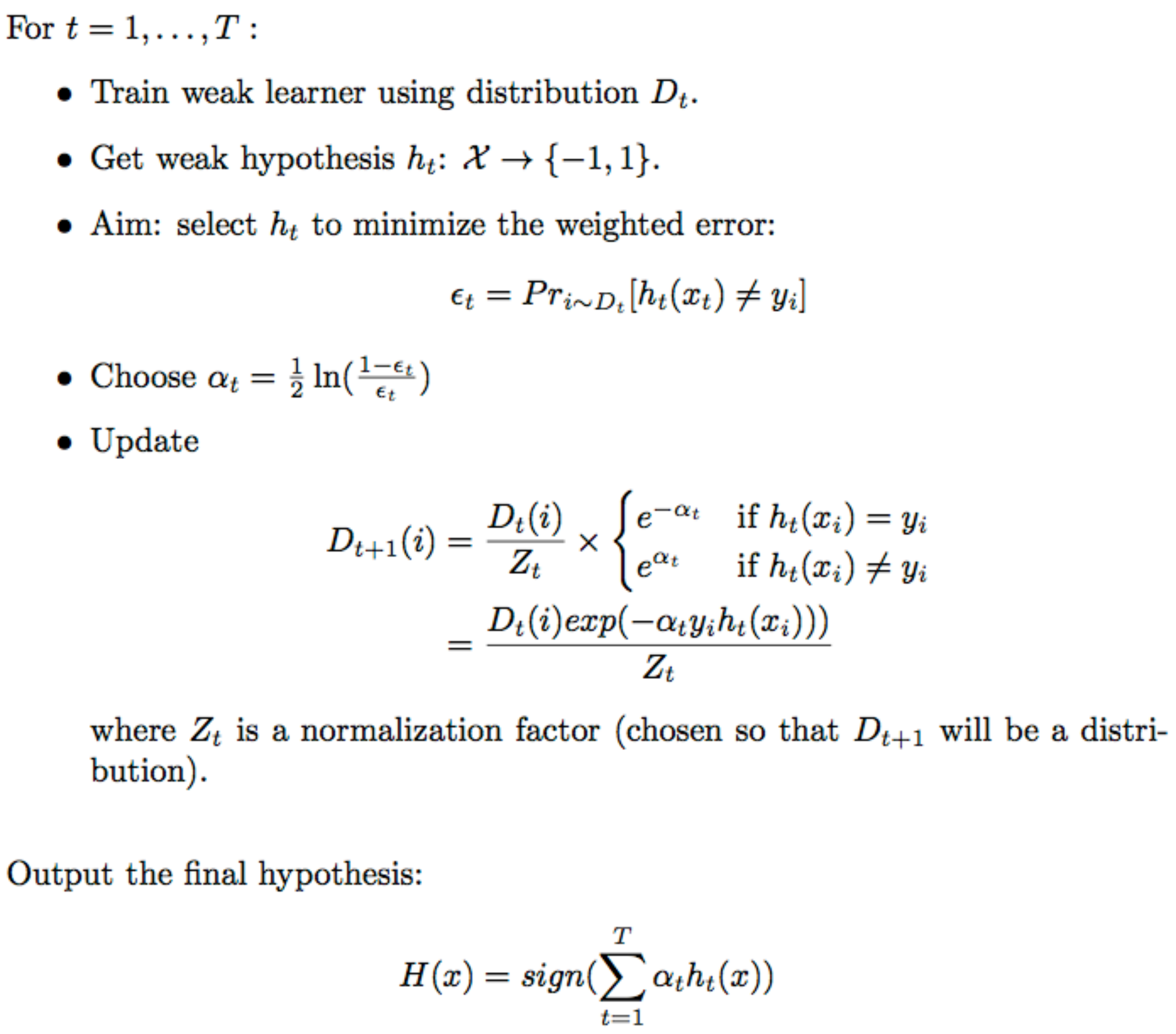
    $$h_t \colon \mathcal{X} \to \{-1, 1\}$$

  - observe error(h,D); compute coefficient

    $$\epsilon_t = Pr_{i \sim D_t}[h_t(x_t) \neq y_i] \qquad \alpha_t = \frac{1}{2}\ln\left(\frac{1-\epsilon_t}{\epsilon_t}\right)$$

  - store weak learner h_t , coefficient $\boldsymbol{\alpha}_t$
  - update Distribution D for next round, emphasizing misclassified points
- AdaBoost final classifier
- $$H(x) = sign\left(\sum_{t=1}^{T} \alpha_t h_t(x)\right)$$

Adaboost Algorithm



For $t = 1, \ldots, T$:

- Train weak learner using distribution D_t.
- Get weak hypothesis h_t: $\mathcal{X} \to \{-1, 1\}$.
- Aim: select h_t to minimize the weighted error:

$$\epsilon_t = Pr_{i \sim D_t}[h_t(x_t) \neq y_i]$$

- Choose $\alpha_t = \frac{1}{2}\ln(\frac{1-\epsilon_t}{\epsilon_t})$
- Update

$$\begin{aligned} D_{t+1}(i) &= \frac{D_t(i)}{Z_t} \times \begin{cases} e^{-\alpha_t} & \text{if } h_t(x_i) = y_i \\ e^{\alpha_t} & \text{if } h_t(x_i) \neq y_i \end{cases} \\ &= \frac{D_t(i) exp(-\alpha_t y_i h_t(x_i)))}{Z_t} \end{aligned}$$

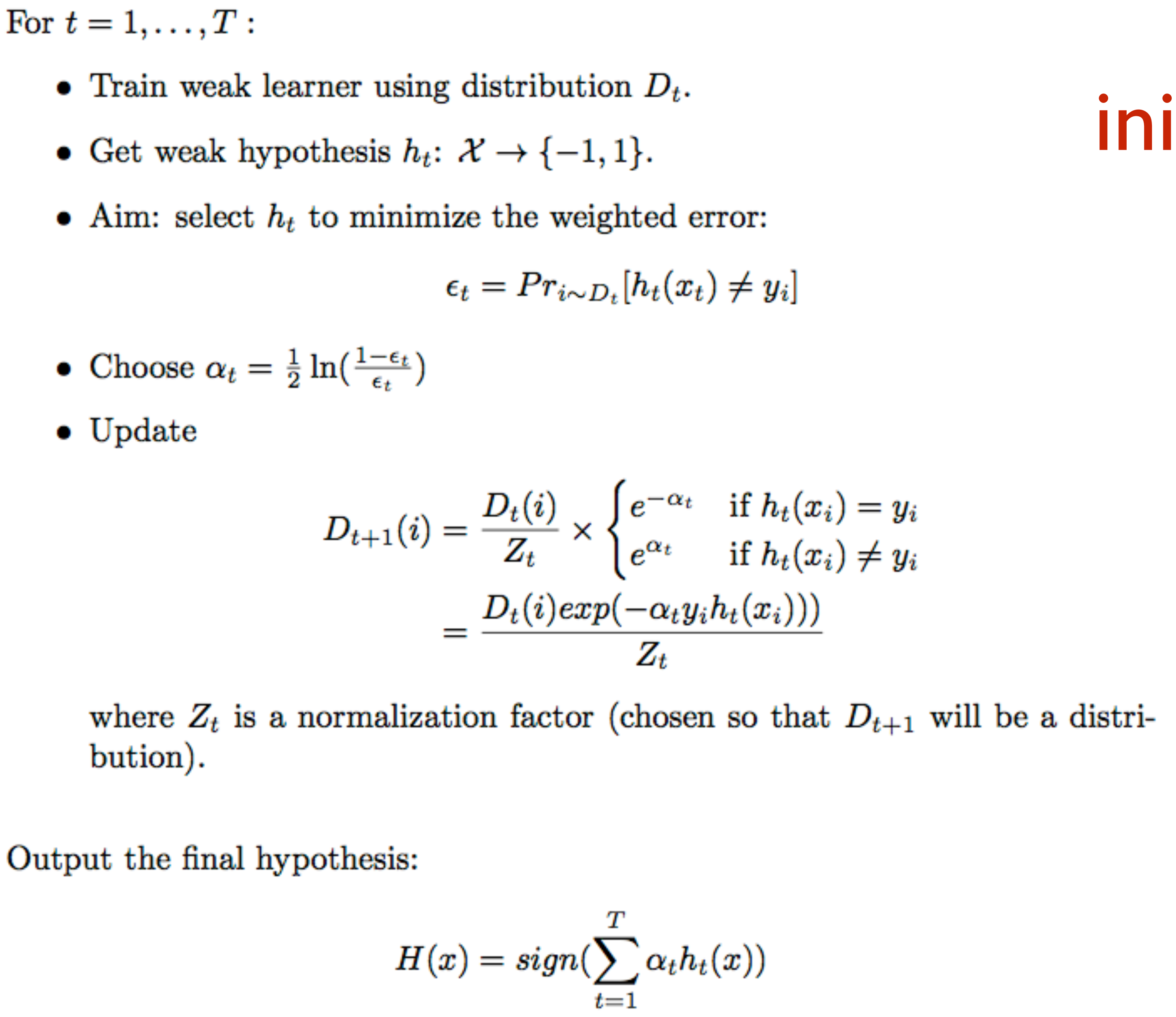
where Z_t is a normalization factor (chosen so that D_{t+1} will be a distribution).

Output the final hypothesis:

$$H(x) = sign(\sum_{t=1}^{T} \alpha_t h_t(x))$$

Adaboost Algorithm

init setup



For $t = 1, \ldots, T$:

- Train weak learner using distribution D_t.
- Get weak hypothesis h_t: $\mathcal{X} \to \{-1, 1\}$.
- Aim: select h_t to minimize the weighted error:

$$\epsilon_t = Pr_{i \sim D_t}[h_t(x_t) \neq y_i]$$

- Choose $\alpha_t = \frac{1}{2} \ln(\frac{1-\epsilon_t}{\epsilon_t})$
- Update

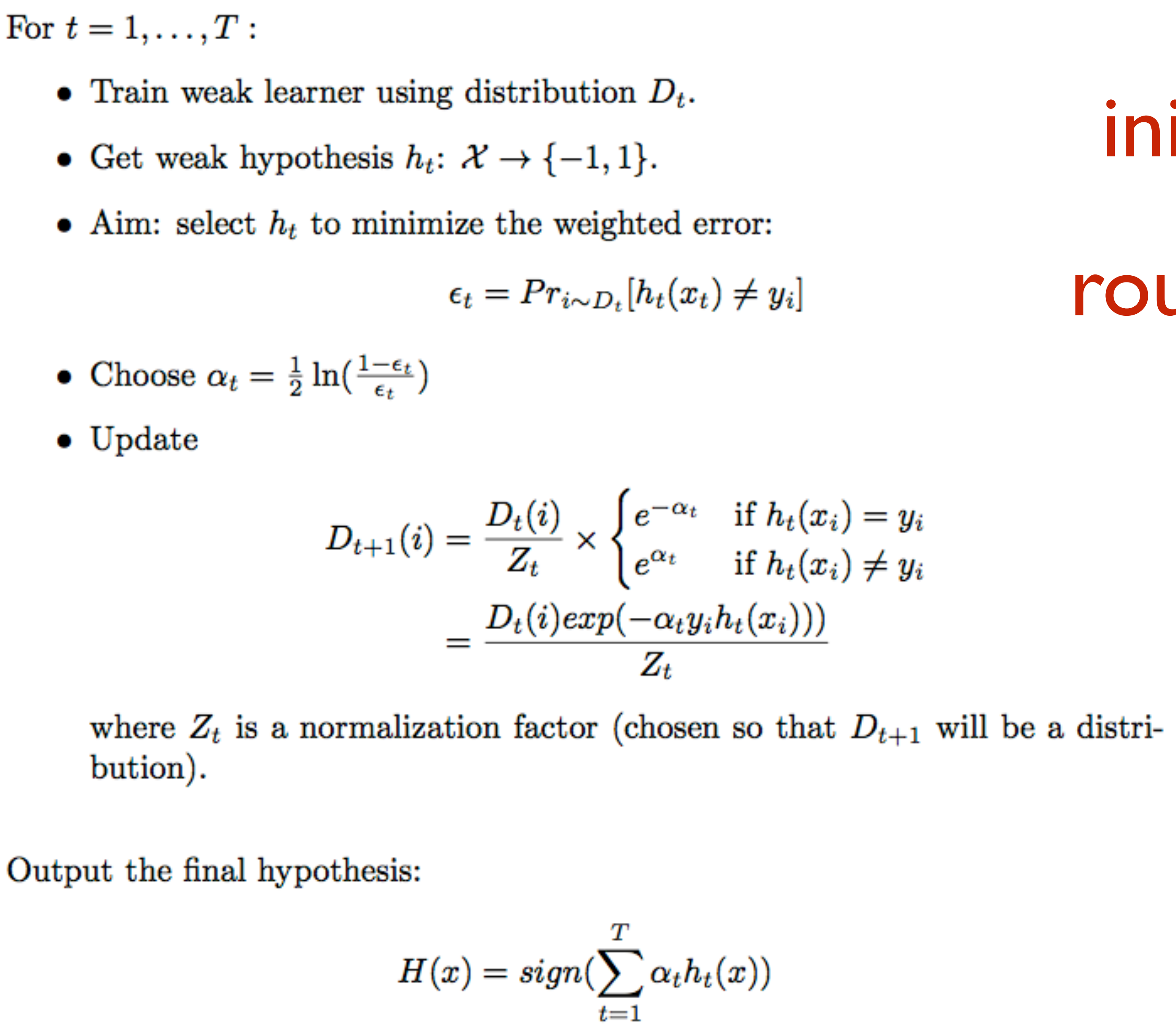
$$\begin{aligned} D_{t+1}(i) &= \frac{D_t(i)}{Z_t} \times \begin{cases} e^{-\alpha_t} & \text{if } h_t(x_i) = y_i \\ e^{\alpha_t} & \text{if } h_t(x_i) \neq y_i \end{cases} \\ &= \frac{D_t(i) exp(-\alpha_t y_i h_t(x_i)))}{Z_t} \end{aligned}$$

where Z_t is a normalization factor (chosen so that D_{t+1} will be a distribution).

Output the final hypothesis:

$$H(x) = sign(\sum_{t=1}^{T} \alpha_t h_t(x))$$

Adaboost Algorithm



For $t = 1, \ldots, T$:

- Train weak learner using distribution D_t.
- Get weak hypothesis h_t: $\mathcal{X} \rightarrow \{-1, 1\}$.
- Aim: select h_t to minimize the weighted error:

$$\epsilon_t = Pr_{i \sim D_t}[h_t(x_t) \neq y_i]$$

- Choose $\alpha_t = \frac{1}{2} \ln(\frac{1-\epsilon_t}{\epsilon_t})$
- Update

$$\begin{aligned} D_{t+1}(i) &= \frac{D_t(i)}{Z_t} \times \begin{cases} e^{-\alpha_t} & \text{if } h_t(x_i) = y_i \\ e^{\alpha_t} & \text{if } h_t(x_i) \neq y_i \end{cases} \\ &= \frac{D_t(i) exp(-\alpha_t y_i h_t(x_i)))}{Z_t} \end{aligned}$$

where Z_t is a normalization factor (chosen so that D_{t+1} will be a distribution).

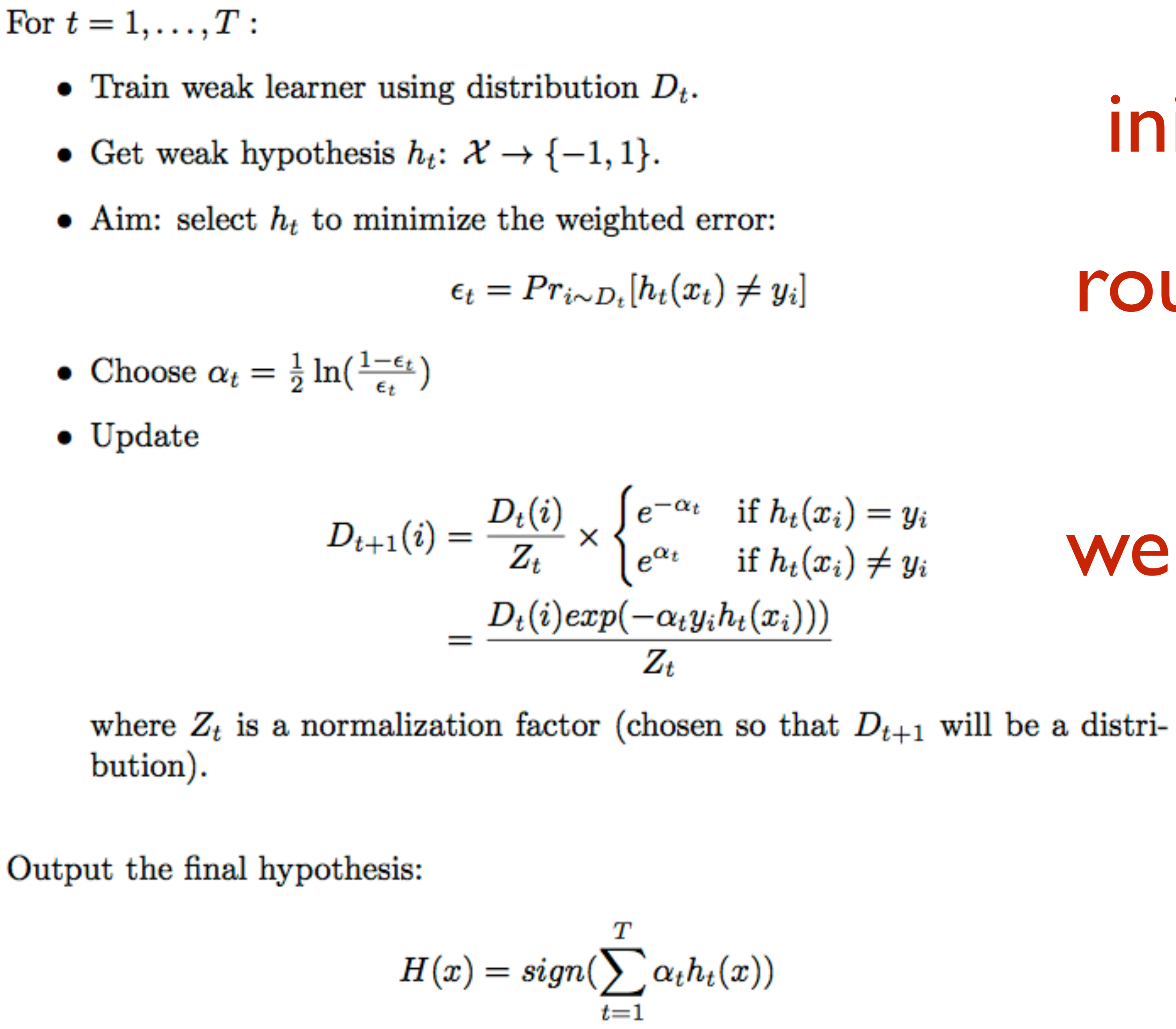
Output the final hypothesis:

$$H(x) = sign(\sum_{t=1}^{T} \alpha_t h_t(x))$$

init setup

round error

Adaboost Algorithm



For $t = 1, \dots, T$:

- Train weak learner using distribution D_t.
- Get weak hypothesis h_t: $\mathcal{X} \to \{-1, 1\}$.
- Aim: select h_t to minimize the weighted error:

$$\epsilon_t = Pr_{i \sim D_t}[h_t(x_t) \neq y_i]$$

- Choose $\alpha_t = \frac{1}{2} \ln(\frac{1-\epsilon_t}{\epsilon_t})$
- Update

$$\begin{aligned} D_{t+1}(i) &= \frac{D_t(i)}{Z_t} \times \begin{cases} e^{-\alpha_t} & \text{if } h_t(x_i) = y_i \\ e^{\alpha_t} & \text{if } h_t(x_i) \neq y_i \end{cases} \\ &= \frac{D_t(i) exp(-\alpha_t y_i h_t(x_i)))}{Z_t} \end{aligned}$$

where Z_t is a normalization factor (chosen so that D_{t+1} will be a distribution).

Output the final hypothesis:

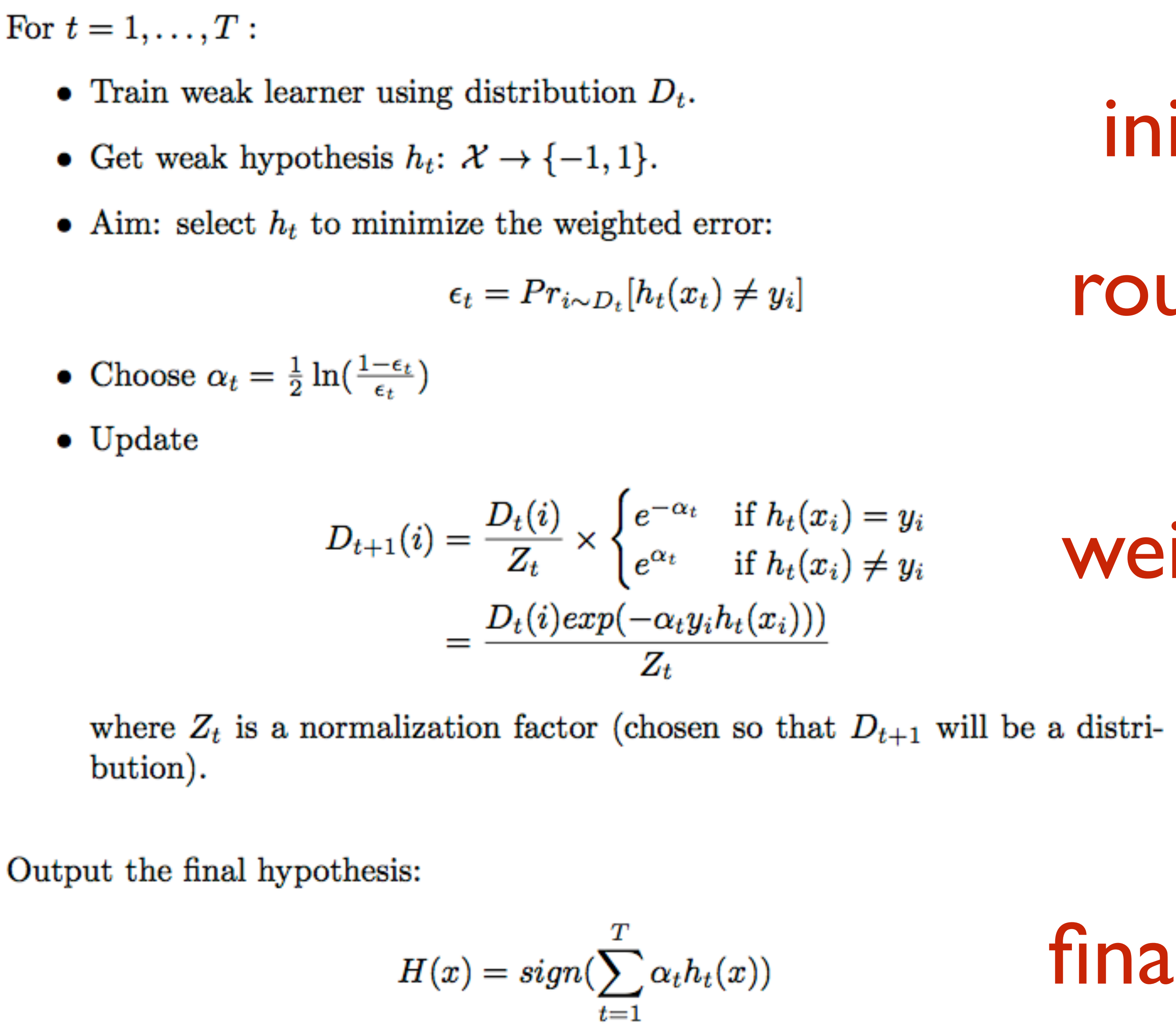
$$H(x) = sign(\sum_{t=1}^{T} \alpha_t h_t(x))$$

init setup

round error

weight update

Adaboost Algorithm



For $t = 1, \ldots, T$:

- Train weak learner using distribution D_t.
- Get weak hypothesis h_t: $\mathcal{X} \rightarrow \{-1, 1\}$.
- Aim: select h_t to minimize the weighted error:

$$\epsilon_t = Pr_{i \sim D_t}[h_t(x_t) \neq y_i]$$

- Choose $\alpha_t = \frac{1}{2} \ln(\frac{1-\epsilon_t}{\epsilon_t})$
- Update

$$D_{t+1}(i) = \frac{D_t(i)}{Z_t} \times \begin{cases} e^{-\alpha_t} & \text{if } h_t(x_i) = y_i \\ e^{\alpha_t} & \text{if } h_t(x_i) \neq y_i \end{cases}$$
$$= \frac{D_t(i) exp(-\alpha_t y_i h_t(x_i)))}{Z_t}$$

where Z_t is a normalization factor (chosen so that D_{t+1} will be a distribution).

Output the final hypothesis:

$$H(x) = sign(\sum_{t=1}^{T} \alpha_t h_t(x))$$

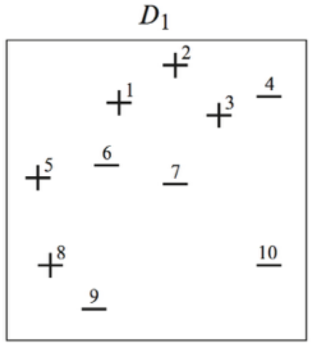
init setup

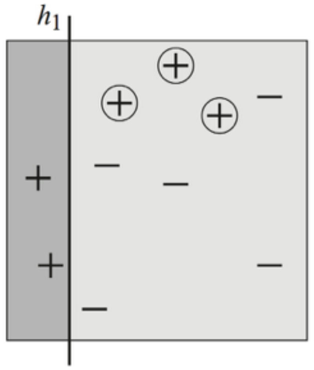
round error

weight update

final classifier

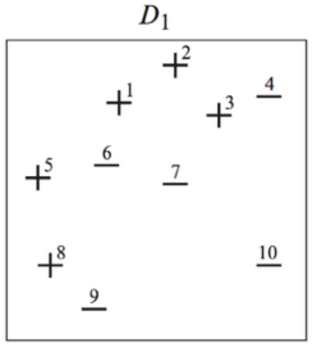
Adaboost: an example

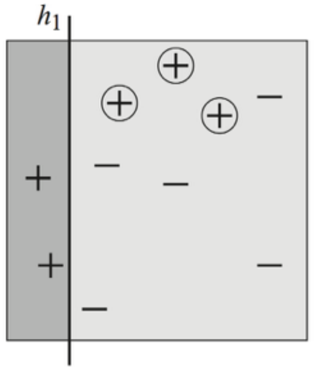




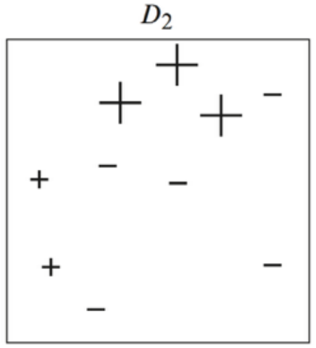
	1	2	3	4	5	6	7	8	9	10	
$D_1(i)$	$\underline{0.10}$	$\underline{0.10}$	$\underline{0.10}$	0.10	0.10	0.10	0.10	0.10	0.10	0.10	$\epsilon_1 = 0.30, \alpha_1 \approx 0.42$
$e^{-\alpha_1 y_i h_1(x_i)}$	1.53	1.53	1.53	0.65	0.65	0.65	0.65	0.65	0.65	0.65	
$D_1(i)\ e^{-\alpha_1 y_i h_1(x_i)}$	0.15	0.15	0.15	0.07	0.07	0.07	0.07	0.07	0.07	0.07	$Z_1 \approx 0.92$

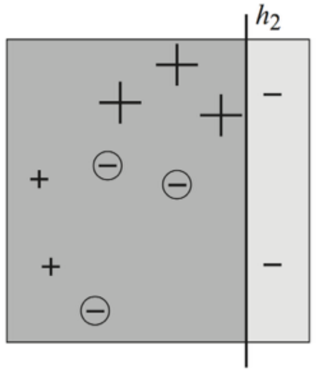
Adaboost: an example





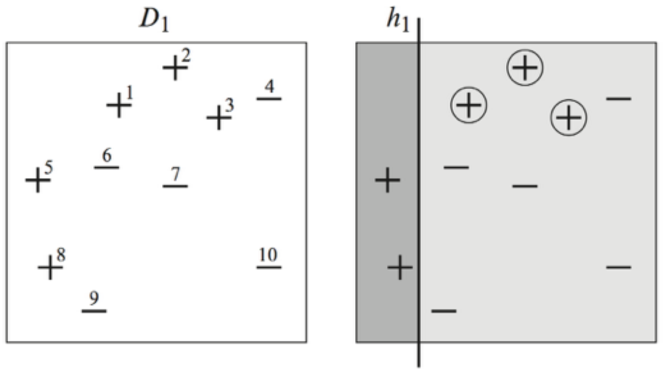
	1	2	3	4	5	6	7	8	9	10	
$D_1(i)$	$\underline{0.10}$	$\underline{0.10}$	$\underline{0.10}$	0.10	0.10	0.10	0.10	0.10	0.10	0.10	$\epsilon_1 = 0.30, \alpha_1 \approx 0.42$
$e^{-\alpha_1 y_i h_1(x_i)}$	1.53	1.53	1.53	0.65	0.65	0.65	0.65	0.65	0.65	0.65	
$D_1(i)\ e^{-\alpha_1 y_i h_1(x_i)}$	0.15	0.15	0.15	0.07	0.07	0.07	0.07	0.07	0.07	0.07	$Z_1 \approx 0.92$



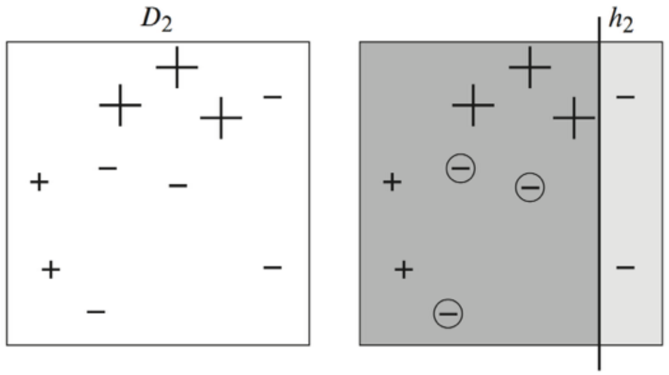


$D_2(i)$	0.17	0.17	0.17	0.07	0.07	$\underline{0.07}$	$\underline{0.07}$	0.07	$\underline{0.07}$	0.07	$\epsilon_2 \approx 0.21, \alpha_2 \approx 0.65$
$e^{-\alpha_2 y_i h_2(x_i)}$	0.52	0.52	0.52	0.52	0.52	1.91	1.91	0.52	1.91	0.52	
$D_2(i)\ e^{-\alpha_2 y_i h_2(x_i)}$	0.09	0.09	0.09	0.04	0.04	0.14	0.14	0.04	0.14	0.04	$Z_2 \approx 0.82$

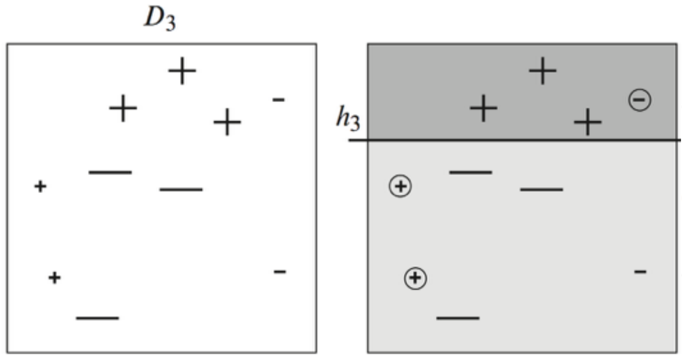
Adaboost: an example



	1	2	3	4	5	6	7	8	9	10	
$D_1(i)$	$\underline{0.10}$	$\underline{0.10}$	$\underline{0.10}$	0.10	0.10	0.10	0.10	0.10	0.10	0.10	$\epsilon_1 = 0.30, \alpha_1 \approx 0.42$
$e^{-\alpha_1 y_i h_1(x_i)}$	1.53	1.53	1.53	0.65	0.65	0.65	0.65	0.65	0.65	0.65	
$D_1(i)\ e^{-\alpha_1 y_i h_1(x_i)}$	0.15	0.15	0.15	0.07	0.07	0.07	0.07	0.07	0.07	0.07	$Z_1 \approx 0.92$

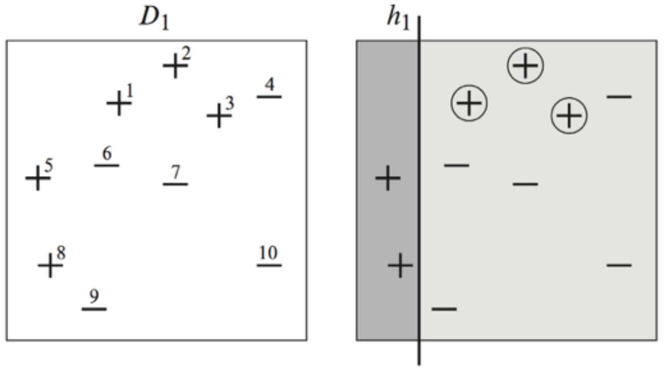


$D_2(i)$	0.17	0.17	0.17	0.07	0.07	$\underline{0.07}$	$\underline{0.07}$	0.07	$\underline{0.07}$	0.07	$\epsilon_2 \approx 0.21, \alpha_2 \approx 0.65$
$e^{-\alpha_2 y_i h_2(x_i)}$	0.52	0.52	0.52	0.52	0.52	1.91	1.91	0.52	1.91	0.52	
$D_2(i)\ e^{-\alpha_2 y_i h_2(x_i)}$	0.09	0.09	0.09	0.04	0.04	0.14	0.14	0.04	0.14	0.04	$Z_2 \approx 0.82$

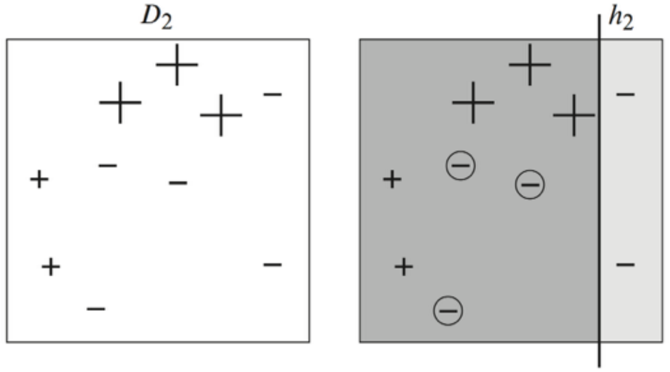


$D_3(i)$	0.11	0.11	0.11	$\underline{0.05}$	$\underline{0.05}$	0.17	0.17	$\underline{0.05}$	0.17	0.05	$\epsilon_3 \approx 0.14, \alpha_3 \approx 0.92$
$e^{-\alpha_3 y_i h_3(x_i)}$	0.40	0.40	0.40	2.52	2.52	0.40	0.40	2.52	0.40	0.40	
$D_3(i)\ e^{-\alpha_3 y_i h_3(x_i)}$	0.04	0.04	0.04	0.11	0.11	0.07	0.07	0.11	0.07	0.02	$Z_3 \approx 0.69$

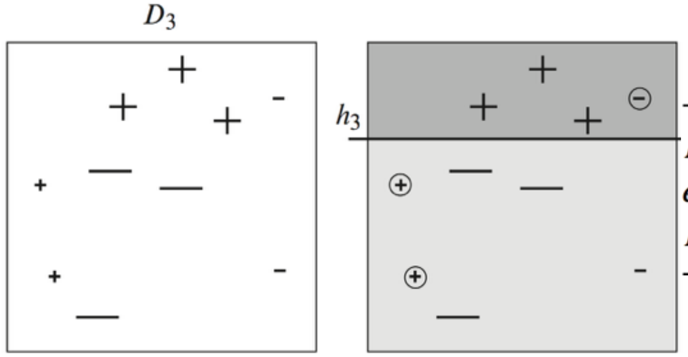
Adaboost: an example



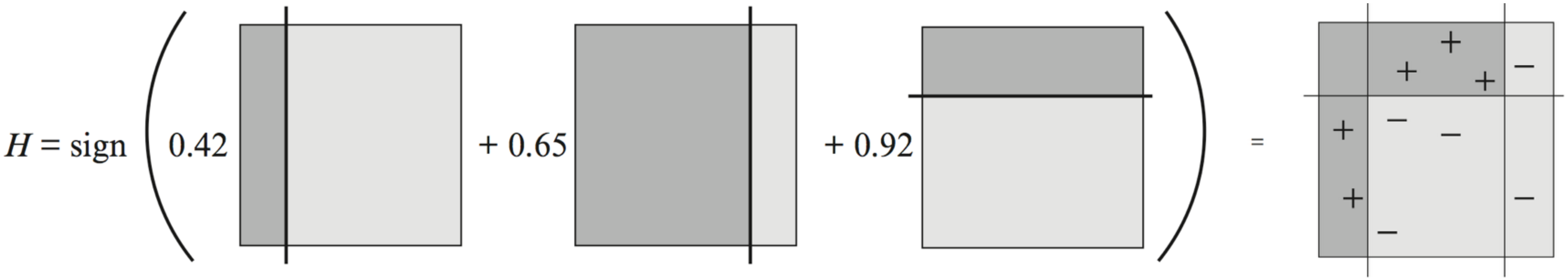
	1	2	3	4	5	6	7	8	9	10	
$D_1(i)$	$\underline{0.10}$	$\underline{0.10}$	$\underline{0.10}$	0.10	0.10	0.10	0.10	0.10	0.10	0.10	$\epsilon_1 = 0.30, \alpha_1 \approx 0.42$
$e^{-\alpha_1 y_i h_1(x_i)}$	1.53	1.53	1.53	0.65	0.65	0.65	0.65	0.65	0.65	0.65	
$D_1(i)\ e^{-\alpha_1 y_i h_1(x_i)}$	0.15	0.15	0.15	0.07	0.07	0.07	0.07	0.07	0.07	0.07	$Z_1 \approx 0.92$



$D_2(i)$	0.17	0.17	0.17	0.07	0.07	$\underline{0.07}$	$\underline{0.07}$	0.07	$\underline{0.07}$	0.07	$\epsilon_2 \approx 0.21, \alpha_2 \approx 0.65$
$e^{-\alpha_2 y_i h_2(x_i)}$	0.52	0.52	0.52	0.52	0.52	1.91	1.91	0.52	1.91	0.52	
$D_2(i)\ e^{-\alpha_2 y_i h_2(x_i)}$	0.09	0.09	0.09	0.04	0.04	0.14	0.14	0.04	0.14	0.04	$Z_2 \approx 0.82$



$D_3(i)$	0.11	0.11	0.11	$\underline{0.05}$	$\underline{0.05}$	0.17	0.17	$\underline{0.05}$	0.17	0.05	$\epsilon_3 \approx 0.14, \alpha_3 \approx 0.92$
$e^{-\alpha_3 y_i h_3(x_i)}$	0.40	0.40	0.40	2.52	2.52	0.40	0.40	2.52	0.40	0.40	
$D_3(i)\ e^{-\alpha_3 y_i h_3(x_i)}$	0.04	0.04	0.04	0.11	0.11	0.07	0.07	0.11	0.07	0.02	$Z_3 \approx 0.69$



Adaboost Training error

Let $\gamma = \frac{1}{2} - \epsilon_t$, and let D_1 be an arbitrary initial distribution over the training set. It can be shown [7] that the weighted training error of the combined classifier H with respect to D_1 is bounded as

$$Pr_{i \sim D_1}[H(x_i) \neq y_i] \leq \prod_{t=1}^{T} \sqrt{1 - 4\gamma_t^2} \leq exp(-2 \sum_{t=1}^{T} \gamma_t^2)$$

Adaboost Training error

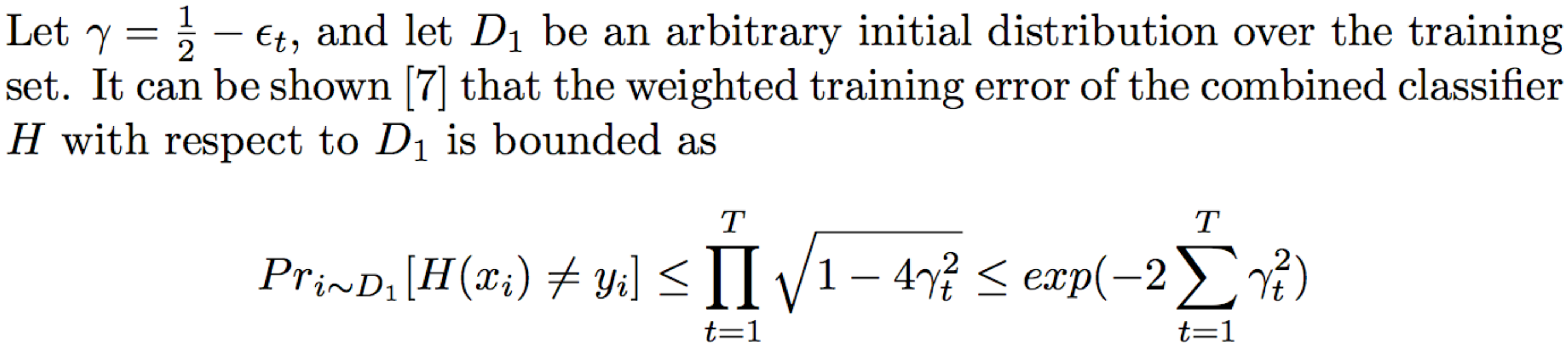
Let $\gamma = \frac{1}{2} - \epsilon_t$, and let D_1 be an arbitrary initial distribution over the training set. It can be shown [7] that the weighted training error of the combined classifier H with respect to D_1 is bounded as

$$Pr_{i \sim D_1}[H(x_i) \neq y_i] \leq \prod_{t=1}^{T} \sqrt{1 - 4\gamma_t^2} \leq exp(-2\sum_{t=1}^{T} \gamma_t^2)$$

There are two possibilities for ending AdaBoost training

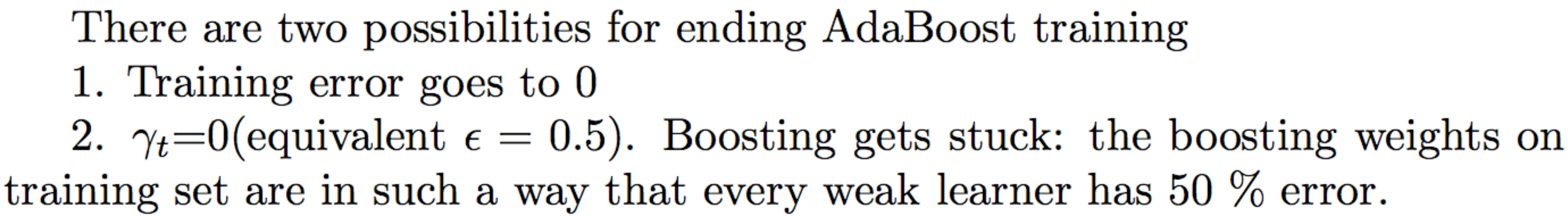
1. Training error goes to 0
2. γ_t=0(equivalent $\epsilon = 0.5$). Boosting gets stuck: the boosting weights on training set are in such a way that every weak learner has 50 % error.

Adaboost Training error



Let $\gamma = \frac{1}{2} - \epsilon_t$, and let D_1 be an arbitrary initial distribution over the training set. It can be shown [7] that the weighted training error of the combined classifier H with respect to D_1 is bounded as

$$Pr_{i\sim D_1}[H(x_i) \neq y_i] \leq \prod_{t=1}^{T} \sqrt{1 - 4\gamma_t^2} \leq exp(-2\sum_{t=1}^{T} \gamma_t^2)$$



There are two possibilities for ending AdaBoost training

1. Training error goes to 0

2. γ_t=0(equivalent $\epsilon = 0.5$). Boosting gets stuck: the boosting weights on training set are in such a way that every weak learner has 50 % error.

- comments: in practice, we usually stop boosting after certain iterations to both save time and prevent overfitting

Adaboost Training error

Proof: Define $F(x) = \sum_{t=1}^{T} \alpha_t h_t(x)$. We have

$$\begin{aligned} D_{T+1}(i) &= D_1(i) \times \frac{e^{-y_i \alpha_1 h_1(x_i)}}{Z_1} \times \cdots \times \frac{e^{-y_i \alpha_T h_T(x_i)}}{Z_T} \\ &= \frac{D_1(i) exp(-y_i \sum_{t=1}^{T} \alpha_t h_t(x_i))}{\prod_{t=1}^{T} Z_t} \\ &= \frac{D_1(i) exp(-y_i F(x_i))}{\prod_{t=1}^{T} Z_t} \end{aligned}$$

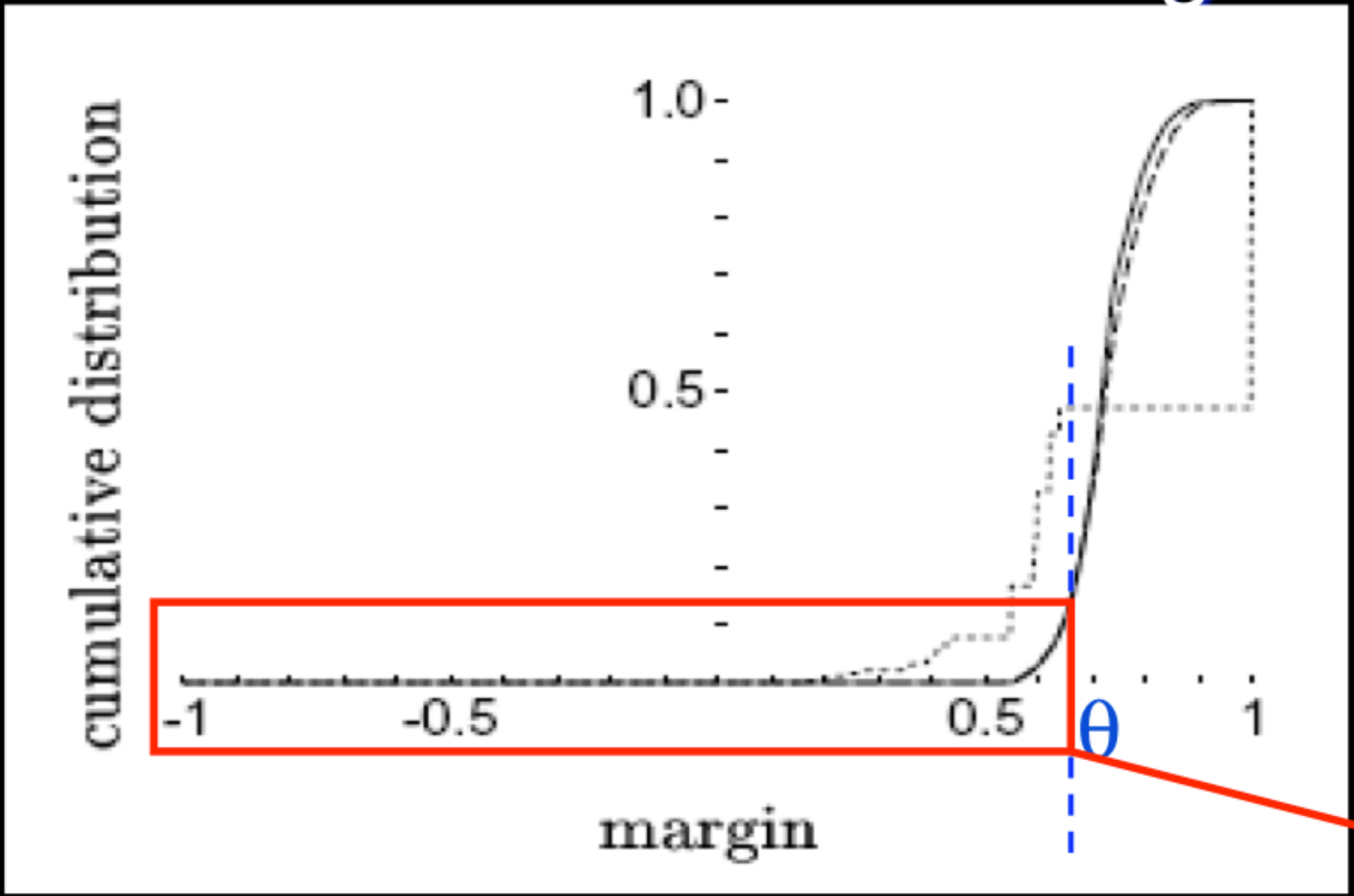
$$\begin{aligned} Pr_{i \sim D_1}[H(x_i) \neq y_i] &= \sum_{i=1}^{m} D_1(i) \mathbf{1}\{H(x_i) \neq y_i\} \\ &\leq \sum_{i=1}^{m} D_1(i) exp(-y_i F(x_i)) \\ &= \sum_{i=1}^{m} D_{T+1}(i) \prod_{t=1}^{T} Z_t \\ &= \prod_{t=1}^{T} Z_t \end{aligned}$$

Adaboost Training error

$$\begin{aligned}
Z_t &= \sum_{i=1}^{m} D_t(i) e^{-\alpha_t y_i h_t(x_i)} \\
&= \sum_{i: y_i = h_t(x_i)} D_t(i) e^{-\alpha_t} + \sum_{i: y_i \neq h_t(x_i)} D_t(i) e^{\alpha_t} \\
&= e^{-\alpha_t}(1 - \epsilon_t) + e^{\alpha_t} \epsilon_t \\
&= e^{-\alpha_t}(\frac{1}{2} + \gamma_t) + e^{\alpha_t}(\frac{1}{2} - \gamma_t) \\
&= \sqrt{1 - 4\gamma_t^2}
\end{aligned}$$

Boosting and Margin Distribution

- Adaboost maximizes margins



data point $x \in X$;

$$F(x) = \sum_t \alpha_t * h_t(x)$$

$$\text{Margin}(x) = y(x) * \frac{F(x)}{\sum_t \alpha_t} \in [-1,1]$$

- Schapire,Freund,Barlett,Lee '98
  - For any 0<θ<1

training miss-confidence

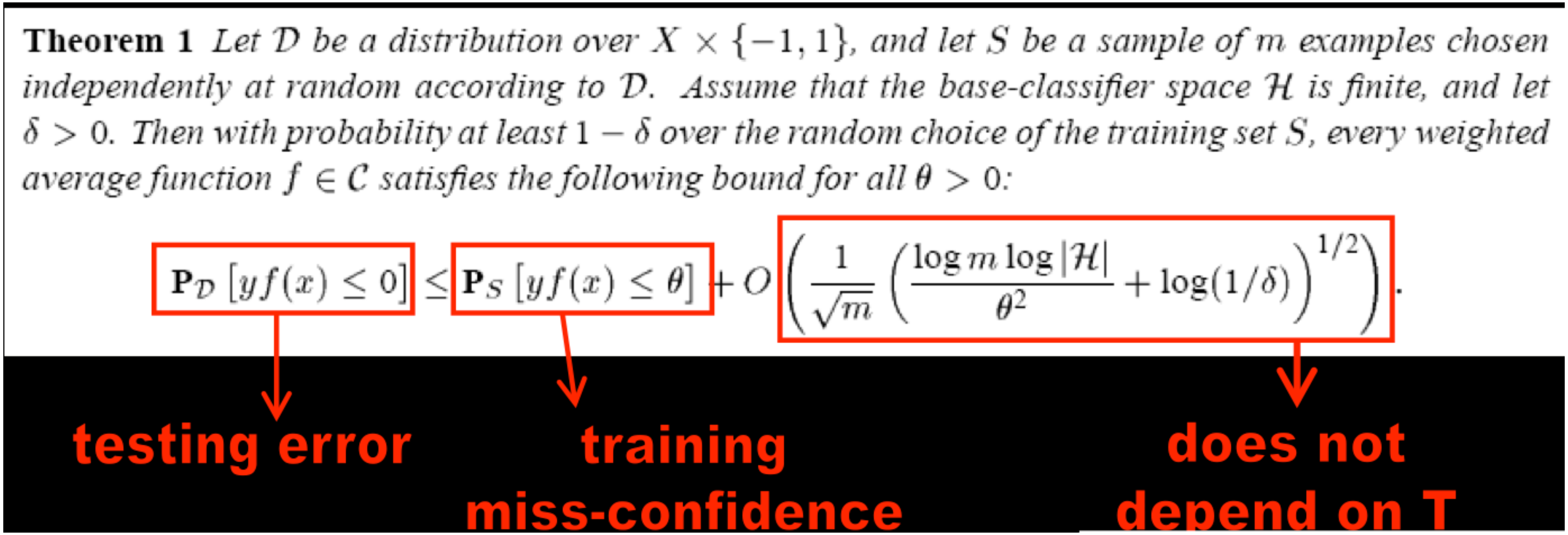
$$\boxed{\mathbf{P}_{(x,y)\sim S}\left[yf(x) \le \theta\right]} \le 2^T \prod_{t=1}^{T} \sqrt{\epsilon_t^{1-\theta}(1-\epsilon_t)^{1+\theta}}.$$

Adaboost testing error based on VC dim

$$\text{testing\_error} \le \text{training\_error} + \mathrm{O}\left(\sqrt{\frac{Td}{m}}\right)$$

- d = VC dim of classifiers (measure of complexity)
- T = number of boosting rounds
  - a loose bound as T can be very large, without decreasing the testing error

Adaboost testing error based on margins



Theorem 1 *Let $\mathcal{D}$ be a distribution over $X \times \{-1,1\}$, and let S be a sample of m examples chosen independently at random according to $\mathcal{D}$. Assume that the base-classifier space $\mathcal{H}$ is finite, and let $\delta > 0$. Then with probability at least $1-\delta$ over the random choice of the training set S, every weighted average function $f \in \mathcal{C}$ satisfies the following bound for all $\theta > 0$:*

$$\underbrace{\boxed{\mathbf{P}_{\mathcal{D}}\left[yf(x) \leq 0\right]}}_{\text{testing error}} \leq \underbrace{\boxed{\mathbf{P}_S\left[yf(x) \leq \theta\right]}}_{\text{training miss-confidence}} + O\underbrace{\boxed{\left(\frac{1}{\sqrt{m}}\left(\frac{\log m \log|\mathcal{H}|}{\theta^2} + \log(1/\delta)\right)^{1/2}\right)}}_{\text{does not depend on T}}.$$

- A better bound for testing error based on margins
- Does not depend on T= number of boosting rounds

Deep decision trees vs Boosted decision stumps

- Deep decision trees and Boosted decision stumps look very similar. Both can easily drive the training error down to 0, and both yield similar decision boundaries. Why does boosted decision stumps often generalize better than deep decision trees?

	Deep Decision Tree	Boosted decision stumps
Partition the space	lines parallel to axis	lines parallel to axis
Decision boundary	zig-zags	zig-zags
Bias	low	low

Deep decision trees vs Boosted decision stumps

	Deep Decision Tree	Boosted decision stumps
Variance	high	low
Representation Power	Each leaf node contains at least one example. The number of examples required to train a constant-leaves decision tree can grow exponentially with the dimension of the input space. Cannot generalize to new variations.	Can generalize to regions not covered by the training set. Have exponentially more efficient power than single decision trees.
voting schema	voting on local tiny regions among data points; more likely to overfit	voting among weak learners. If learners have low complexity, harder to overfit.

Bagging Decision Trees

- Train multiple classifiers, **independently**
- Each classifier = Decision Tree trained on a sampled-with-replacement dataset
- Final prediction: run all classifiers, average their output

Bagging : sampling with replacement

- Trainset of size N; want sampling set of size N

- For i=1:N
  - Randomly select a point Xi from Trainset
  - Do not remove this point so it can be sampled again

- Not all points will be selected
  - selected points expected count ~63%*N
- Some points all be selected multiple times

Bagging Decision Trees VS Boosting

- Both have final prediction as a linear combination of classifiers

- Bagging combination weights are uniform; boosting weights ($\boldsymbol{\alpha}_t$) are a measure of performance for classifier at round

- Bagging has independent classifiers, boosting ones are dependent of each other

- Bagging randomly selects training sets; boosting focuses on most difficult points