

Boosting

Can we make dumb learners smart?

Aarti Singh

Mar 13, 2023

Machine Learning 10-701

Slides Courtesy: Carlos Guestrin, Freund & Schapire



MACHINE LEARNING DEPARTMENT



Why boost weak learners?

Goal: Classify movie review sentiment

“I'm a fan of TV movies in general and this was one of the **good** ones”

“Long, **boring**. Never have I been so glad to see ending credits roll”

“I don't know why I **like** this movie, but I never get tired.”

- **Easy to find “rules of thumb” that are better than random chance.**

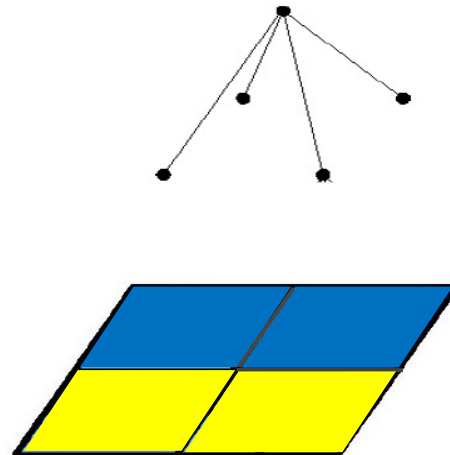
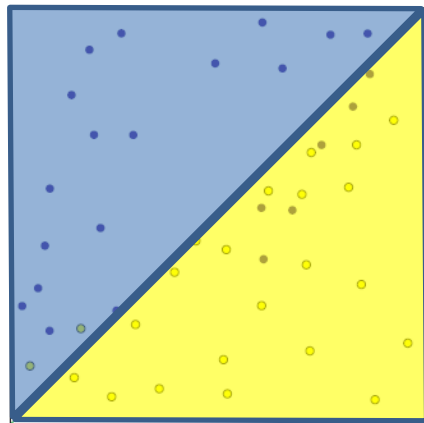
E.g. If ‘good’ occurs in utterance, then predict ‘positive’

- **Hard to find single highly accurate prediction rule.**

e.g. “This movie is terrible but it has some **good** effects”

Fighting the bias-variance tradeoff

- **Simple (a.k.a. weak) learners** e.g., naïve Bayes, logistic regression, decision stumps (or shallow decision trees)



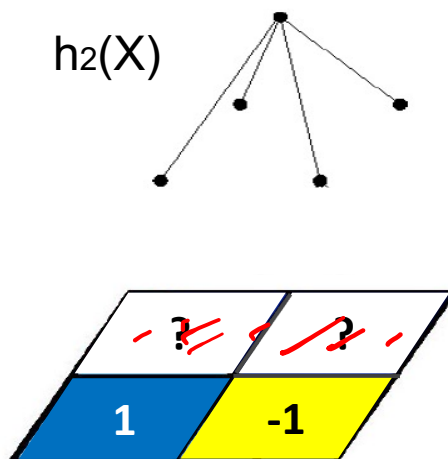
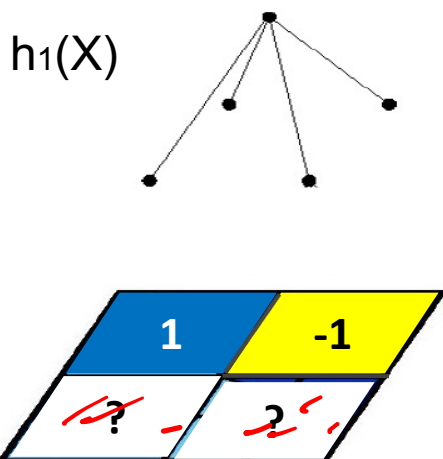
Are good 😊 - don't usually overfit - *var lower*

Are bad 😞 - can't solve hard learning problems - *bias large*

- **Can we make weak learners good???**

Voting (Ensemble Methods)

- Instead of learning a single (weak) classifier, learn **many weak classifiers** that are **good at different parts of the input space**
- **Output class:** (Weighted) vote of each classifier
 - Classifiers that are most “sure” will vote with more conviction
 - Classifiers will be most “sure” about a particular part of the space
 - On average, do better than single classifier!





$$H: X \rightarrow Y (-1,1)$$

$$H(X) = h_1(X) + h_2(X)$$

$$H(X) = \text{sign}\left(\sum_t \alpha_t h_t(X)\right)$$

weights

Voting (Ensemble Methods)

- Instead of learning a single (weak) classifier, learn **many weak classifiers** that are **good at different parts of the input space**
- **Output class:** (Weighted) vote of each classifier
 - Classifiers that are most “sure” will vote with more conviction
 - Classifiers will be most “sure” about a particular part of the space
 - On average, do better than single classifier!
- **But how do you ???**
 - force classifiers h_t to learn about different parts of the input space? 
 - weigh the votes of different classifiers? α_t 

Boosting [Schapire'89]

- **Idea:** given a weak learner, run it multiple times on (reweighted) training data, then let learned classifiers vote
- On each iteration t :
 - weight $D_t(i)$ for each training example i , based on how incorrectly it was classified ✓
 - Learn a weak hypothesis – h_t ✓ *hypothesis = classifier*
 - A weight for this hypothesis – α_t
- Final classifier:
$$H(X) = \text{sign}(\sum \alpha_t h_t(X))$$

- **Practically useful**
- **Theoretically interesting**

Learning from weighted data

- **Consider a weighted dataset**
 - $D(i)$ – weight of i th training example (\mathbf{x}^i, y^i)
 - Interpretations:
 - i th training example counts as $D(i)$ examples
 - If I were to “resample” data, I would get more samples of “heavier” data points
- **Now, in all calculations, whenever used, i th training example counts as $D(i)$ “examples”**
 - e.g., in MLE redefine $Count(Y=y)$ to be weighted count

Unweighted data

$$Count(Y=y) = \sum_{i=1}^m \mathbf{1}(Y^i=y)$$

Weights $D(i)$

$$Count(Y=y) = \sum_{i=1}^m D(i) \mathbf{1}(Y^i=y)$$

AdaBoost [Freund & Schapire'95]

Given: $(x_1, y_1), \dots, (x_m, y_m)$ where $x_i \in X, y_i \in Y = \{-1, +1\}$

Initialize $D_1(i) = 1/m$. **Initially equal weights**

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

- Train weak learner using distribution D_t . **Naïve bayes, decision stump**

- Get weak classifier $h_t : X \rightarrow \mathbb{R}$.

- Choose $\alpha_t \in \mathbb{R}$. **Magic (+ve)**

- Update:

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

$$= \frac{D_t(i) \exp(-\alpha_t y_i h_t(x_i))}{Z_t}$$

**Increase weight
if wrong on pt i
 $y_i h_t(x_i) = -1 < 0$**

where Z_t is a normalization factor

AdaBoost [Freund & Schapire'95]

Given: $(x_1, y_1), \dots, (x_m, y_m)$ where $x_i \in X, y_i \in Y = \{-1, +1\}$

Initialize $D_1(i) = 1/m$. **Initially equal weights**

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

- Train weak learner using distribution D_t . **Naïve bayes, decision stump**

- Get weak classifier $h_t : X \rightarrow \mathbb{R}$.

- Choose $\alpha_t \in \mathbb{R}$. **Magic (+ve)**

- Update:

$$D_{t+1}(i) = \frac{D_t(i) \exp(-\alpha_t y_i h_t(x_i))}{Z_t}$$

$\sum_i D_{t+1}(i) = 1$ ✓

**Increase weight
if wrong on pt i
 $y_i h_t(x_i) = -1 < 0$**

where Z_t is a normalization factor

$$Z_t = \sum_{i=1}^m D_t(i) \exp(-\alpha_t y_i h_t(x_i))$$

**Weights for all
pts must sum to 1
 $\sum_i D_{t+1}(i) = 1$**

AdaBoost [Freund & Schapire'95]

Given: $(x_1, y_1), \dots, (x_m, y_m)$ where $x_i \in X, y_i \in Y = \{-1, +1\}$

Initialize $D_1(i) = 1/m$. **Initially equal weights**

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

- Train weak learner using distribution D_t . **Naïve bayes, decision stump**

- Get weak classifier $h_t : X \rightarrow \mathbb{R}$.

- Choose $\alpha_t \in \mathbb{R}$. **Magic (+ve)**

- Update:

$$D_{t+1}(i) = \frac{D_t(i) \exp(-\alpha_t y_i h_t(x_i))}{Z_t}$$

**Increase weight
if wrong on pt i
 $y_i h_t(x_i) = -1 < 0$**

where Z_t is a normalization factor

Output the final classifier:

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

What α_t to choose for hypothesis h_t ?

$$H(x) = \text{sign}\left(\sum_t \alpha_t h_t(x)\right)$$

Weight Update Rule: $D_{t+1}(i) = \frac{D_t(i) \exp(-\alpha_t y_i h_t(x_i))}{Z_t}$ ✓

$$\epsilon_t = 0 \Rightarrow \alpha_t = \infty \quad \checkmark$$

$$\epsilon_t = 1 \Rightarrow \alpha_t = -\infty \quad \checkmark$$

$$\epsilon_t = 0.5 \Rightarrow \alpha_t = 0 \quad \checkmark$$

$$\alpha_t = \frac{1}{2} \ln \left(\frac{1 - \epsilon_t}{\epsilon_t} \right)$$

[Freund & Schapire'95]

Weighted training error

$$\epsilon_t = P_{i \sim D_t(i)} [h_t(x^i) \neq y^i] = \sum_{i=1}^m D_t(i) \delta(h_t(x_i) \neq y_i)$$

Does h_t get i^{th} point wrong

$\epsilon_t = 0$ if h_t perfectly classifies all weighted data pts

$$\alpha_t = \infty$$

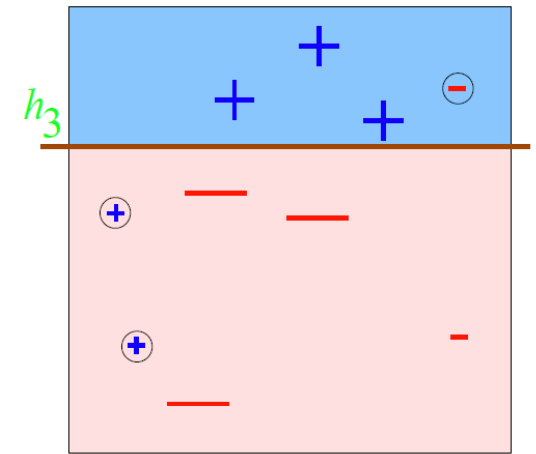
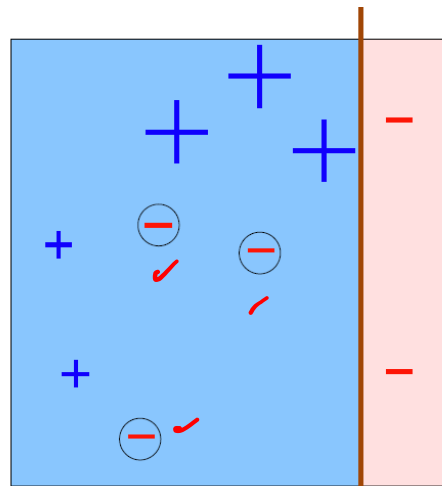
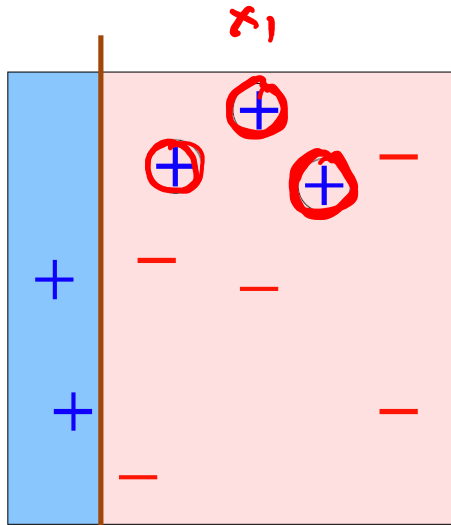
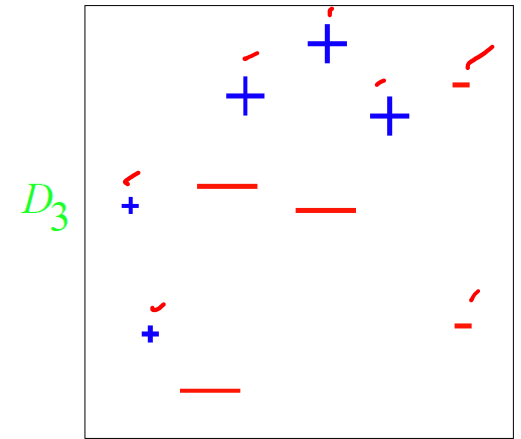
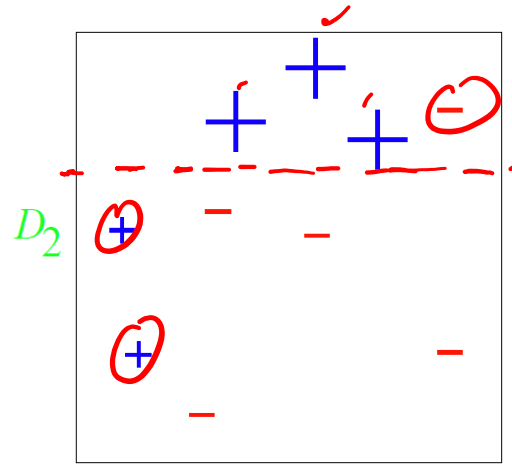
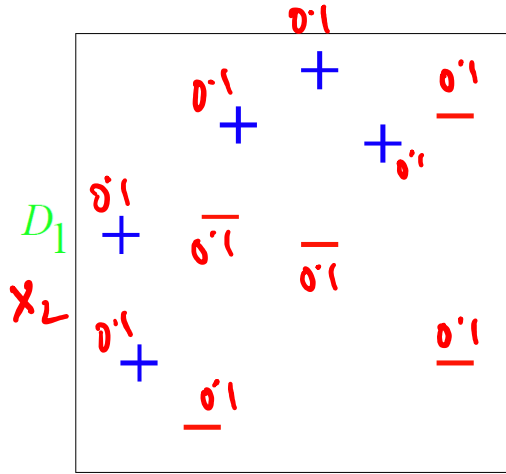
$\epsilon_t = 1$ if h_t perfectly wrong \Rightarrow $-h_t$ perfectly right

$$\alpha_t = -\infty$$

$\epsilon_t = 0.5$

$$\alpha_t = 0$$

Boosting Example (Decision Stumps)



h_1

$$\epsilon_1 = 0.3$$

$$\alpha_1 = \frac{1}{2} \ln \frac{0.7}{0.3} \checkmark$$

ϵ_2

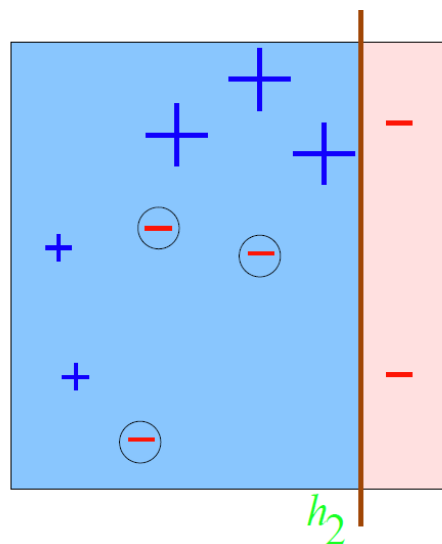
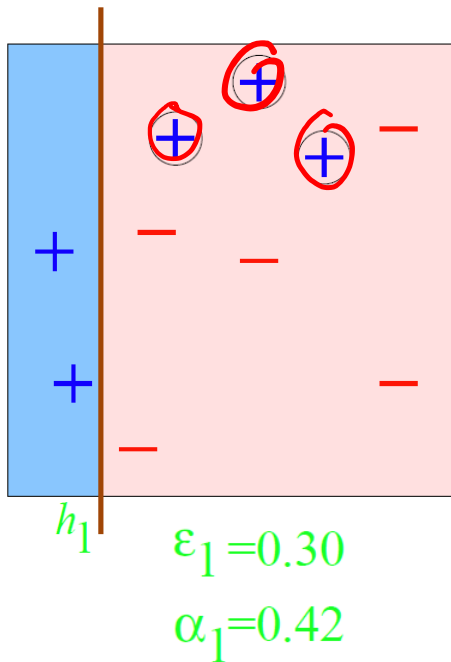
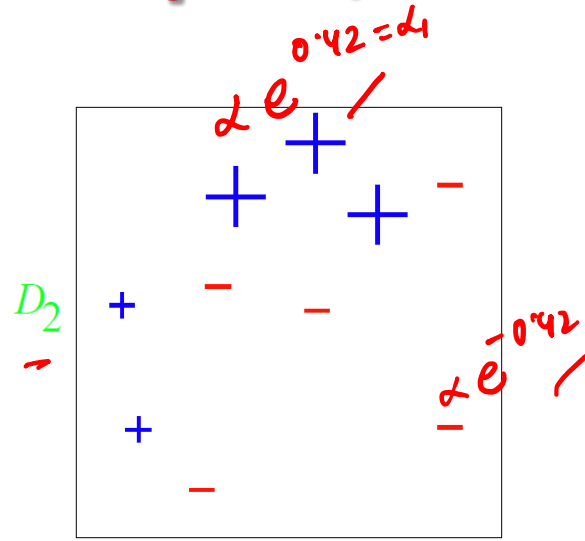
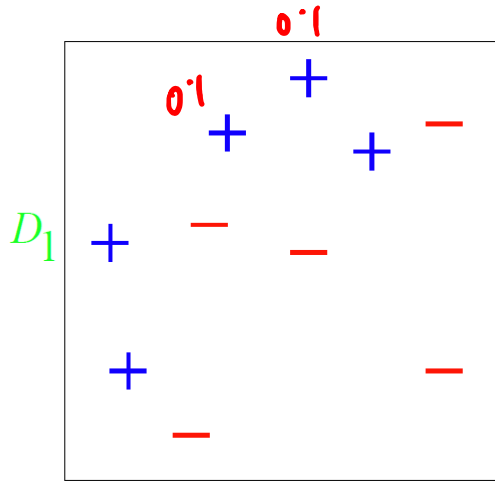
α_2

h_2

ϵ_3

α_3

Boosting Example (Decision Stumps)

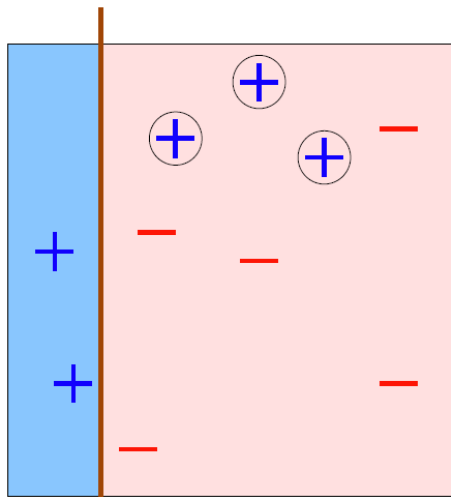
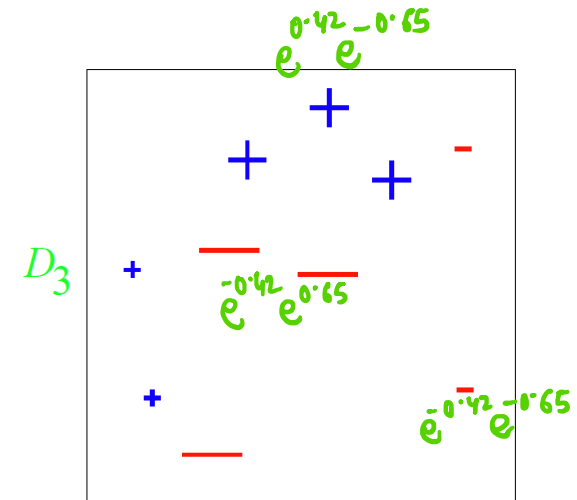
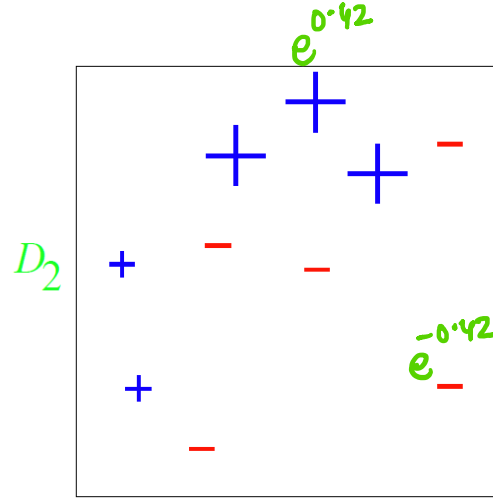
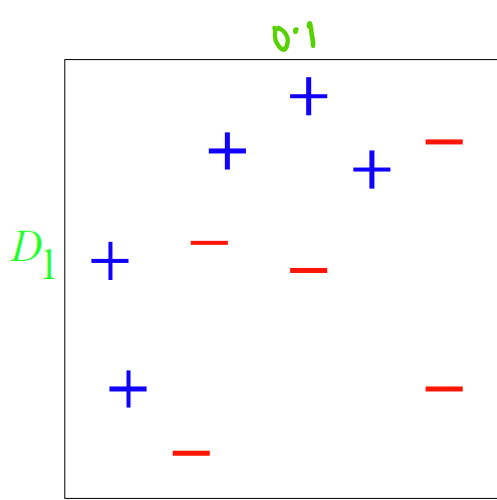


$$\epsilon_2 = \frac{3 \times 0.1 \times e^{-0.42}}{3 \times 0.1 \times e^{-0.42} + 4 \times 0.1 \times e^{-0.42} + 3 \times 0.1 \times e^{0.42}}$$

$$\alpha_2 = \frac{1}{2} \ln \frac{1 - \epsilon_2}{\epsilon_2}$$

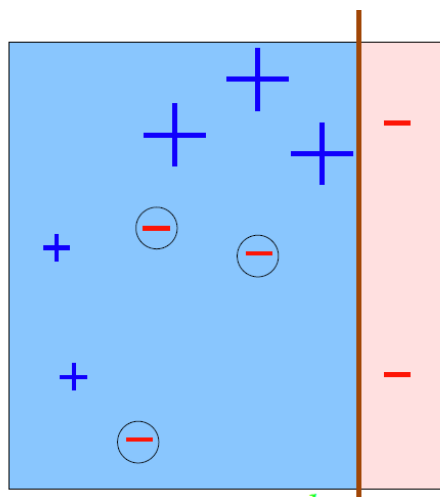
- Poll: What's the error on the weighted training data, ϵ_2 ?

Boosting Example (Decision Stumps)



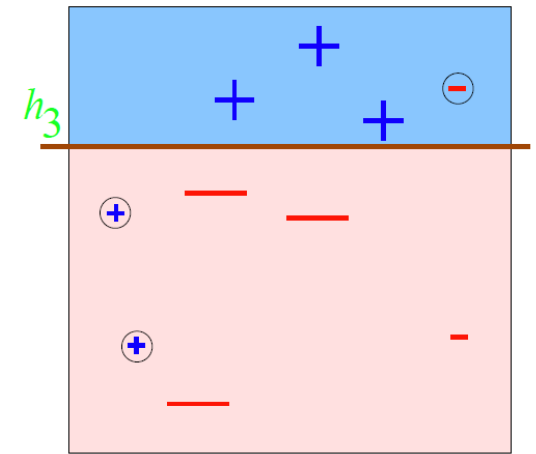
$$\epsilon_1 = 0.30$$

$$\alpha_1 = 0.42$$



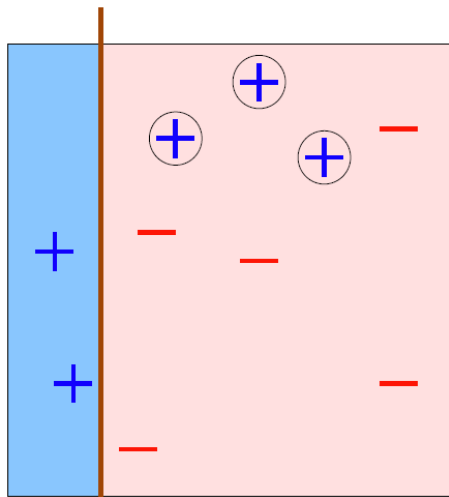
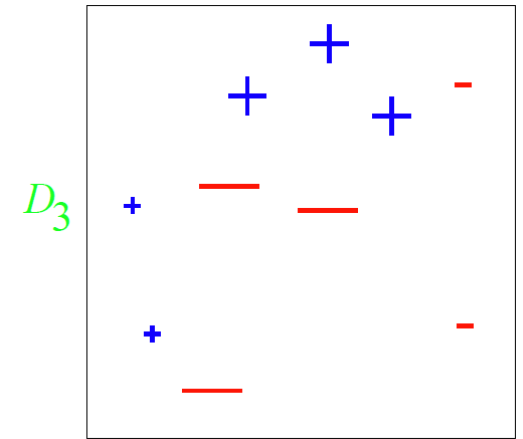
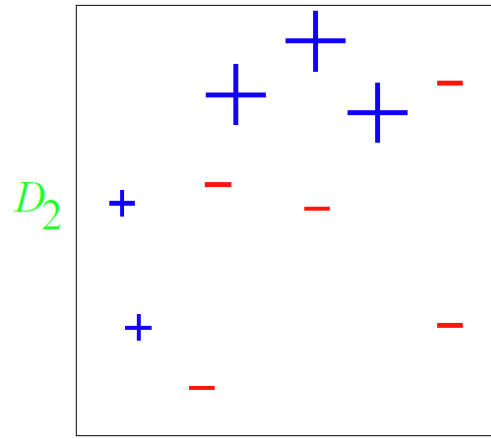
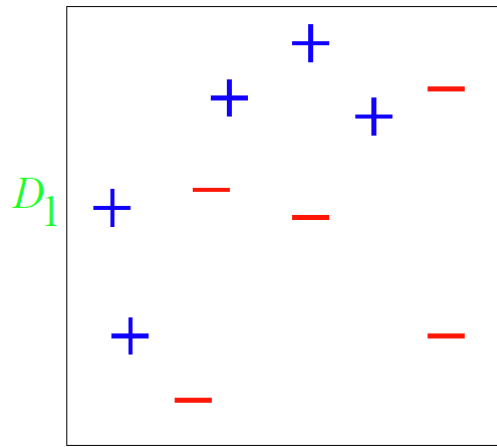
$$\epsilon_2 = 0.21$$

$$\alpha_2 = 0.65$$



$$\epsilon_3 = \frac{3 \times 0.1 \times \exp(-0.42 - 0.65) + 4 \times 0.1 \times \exp(-0.42 - 0.65) + 3 \times 0.1 \times \exp(-0.42 + 0.65) + 3 \times 0.1 \times \exp(0.42 - 0.65)}{14}$$

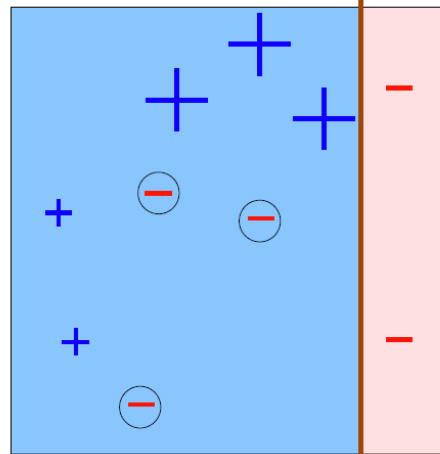
Boosting Example (Decision Stumps)



h_1

$\epsilon_1 = 0.30$

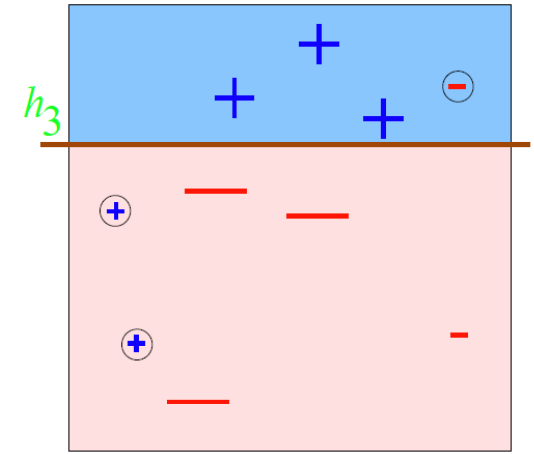
$\alpha_1 = 0.42$



h_2

$\epsilon_2 = 0.21$

$\alpha_2 = 0.65$

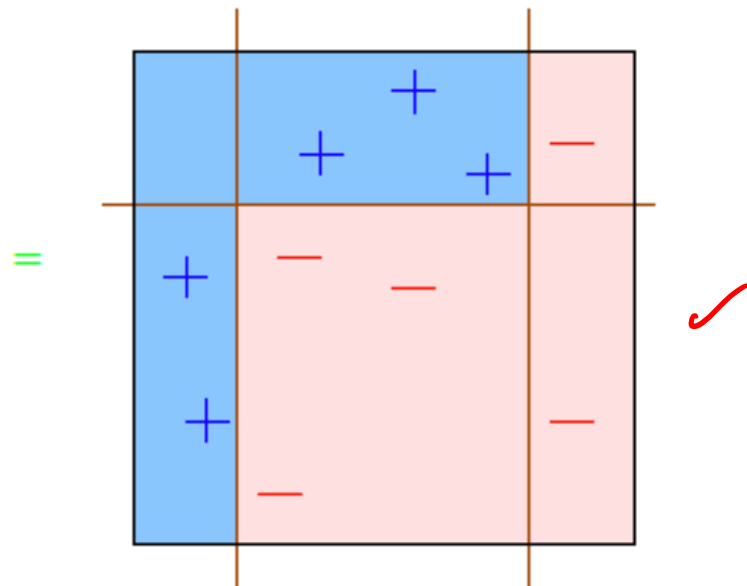
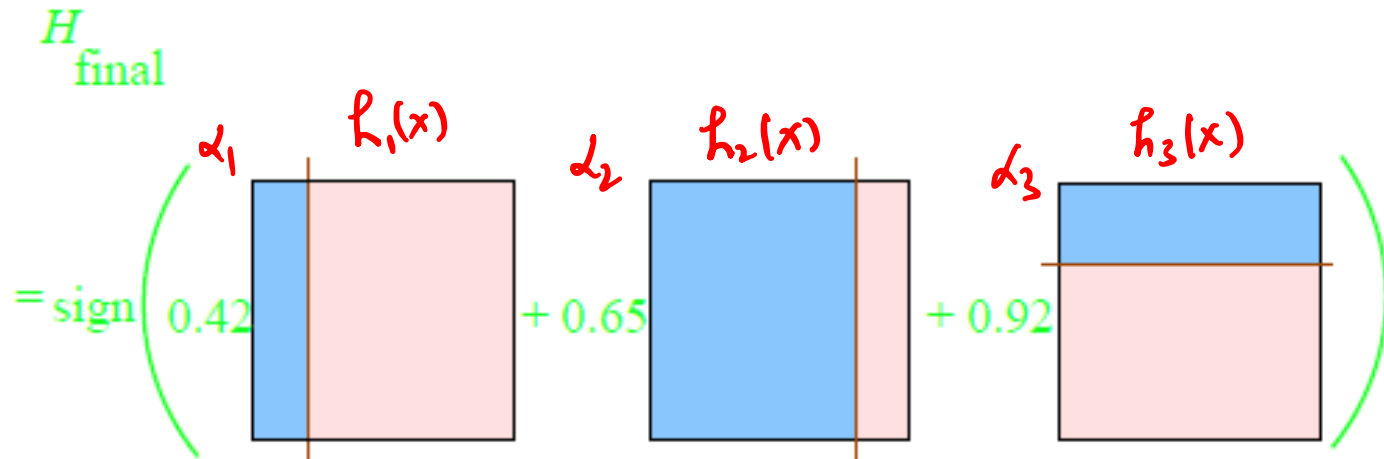


h_3

$\epsilon_3 = 0.14$

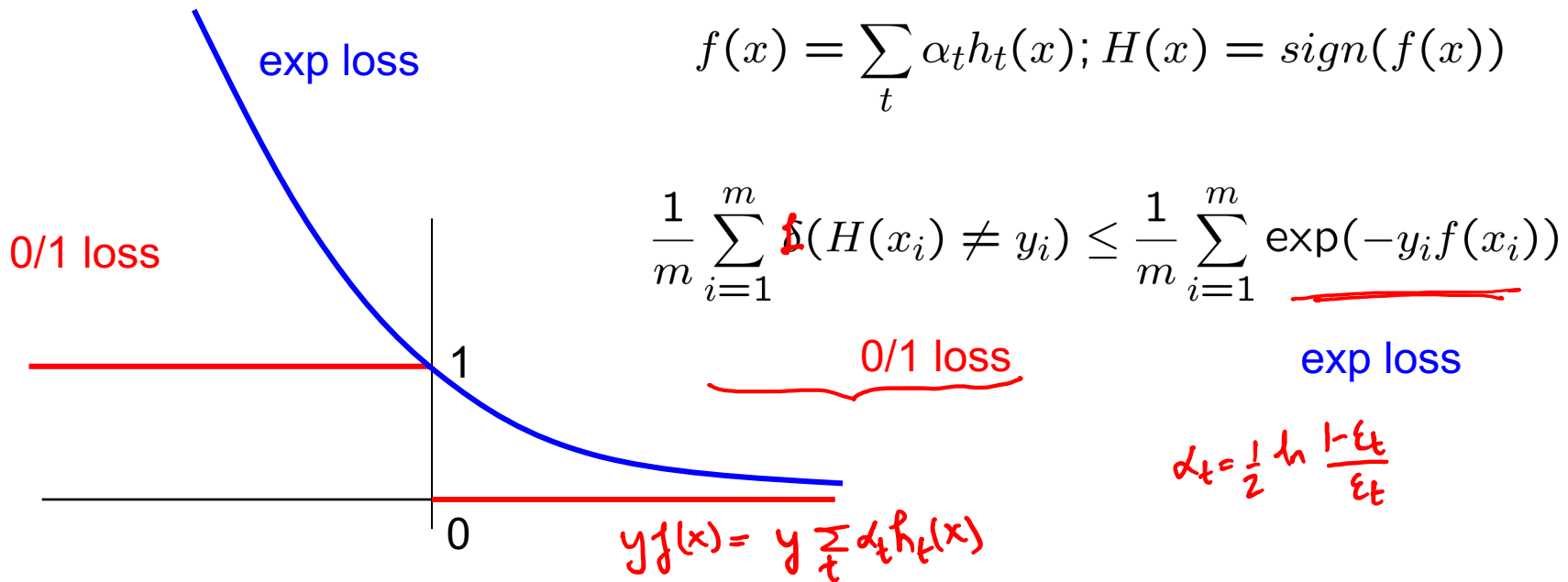
$\alpha_3 = 0.92$

Boosting Example (Decision Stumps)



Analysis for Boosting

- Choice of α_t and hypothesis h_t obtained by coordinate descent on exp loss (convex upper bound on 0/1 loss)



Analysis for Boosting

Analysis reveals:

- If each weak learner h_t is slightly better than random guessing ($\epsilon_t < 0.5$), then training error of AdaBoost decays exponentially fast in number of rounds T .

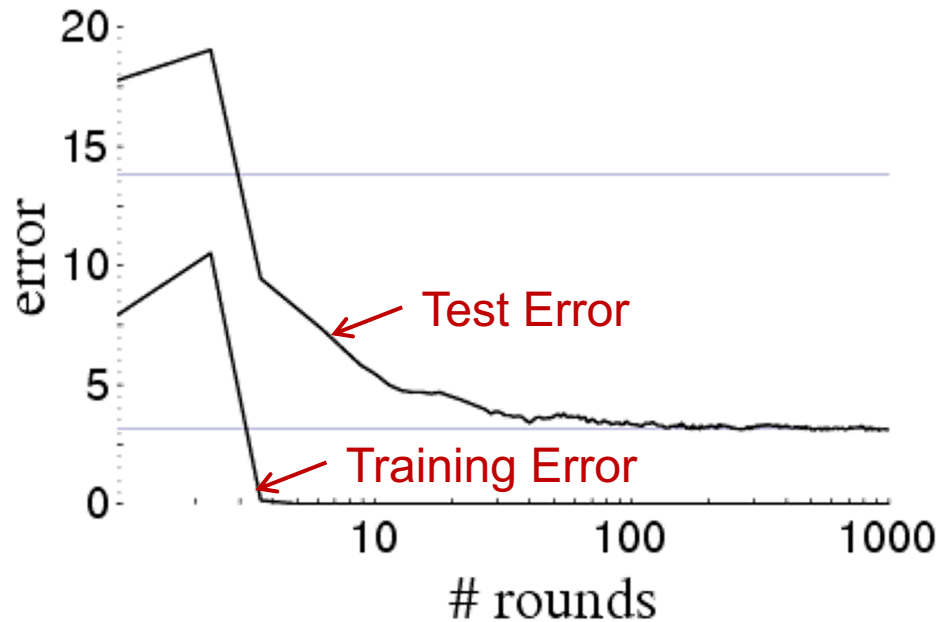
$$\frac{1}{m} \sum_{i=1}^m \delta(H(x_i) \neq y_i) \leq \exp \left(-2 \sum_{t=1}^T (1/2 - \epsilon_t)^2 \right)$$

Training Error

What about test error?

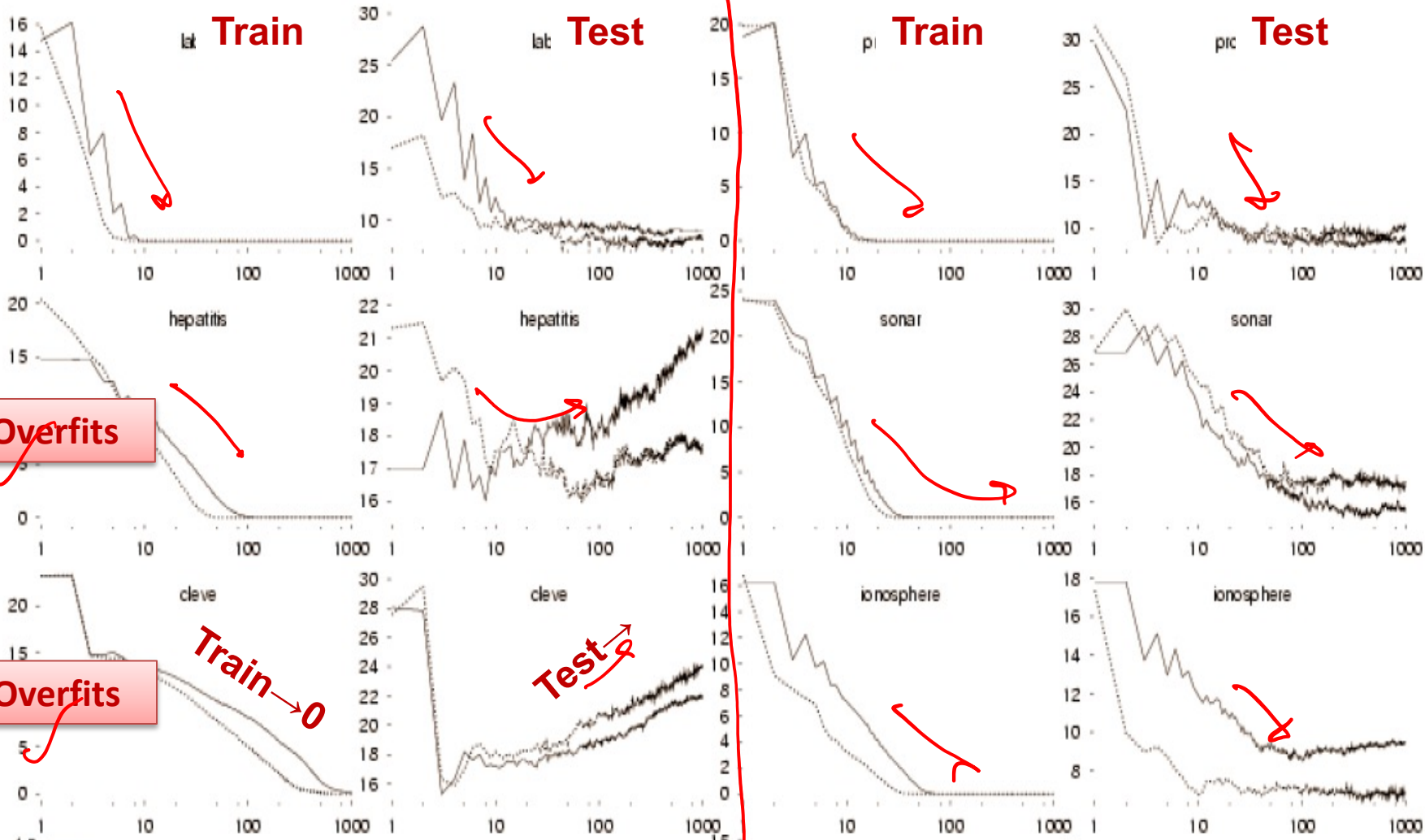
Boosting results – Digit recognition

[Schapire, 1989]



- Boosting often,
 - Robust to overfitting
 - Test set error decreases even after training error is zero
- If classes are well-separated, subsequent weak learners agree and hence more rounds does not necessarily imply that final classifier is getting more complex.

AdaBoost and AdaBoost.MH on Train (left) and Test (right) data from Irvine repository. [Schapire and Singer, ML 1999]



Boosting can overfit if classes not well separated (high label noise) or weak learners are too complex.