Announcements

Assignments

- HW10 (programming + “written”)
  - Due Thu 4/30, 11:59 pm

Final Exam

- Stay tuned to Piazza for details
- Date: Mon 5/11, 5:30 – 8:30 pm
- Format: “online assignment” in Gradescope
- Practice exam: Out later this week
- Recitation this Friday: Review session

Announcements

Course Evaluation Survey

- https://cmu.smartevals.com/
- Take time now to fill this out please

Piazza Poll 1

Did you fill out the FCE survey?

A. Yes

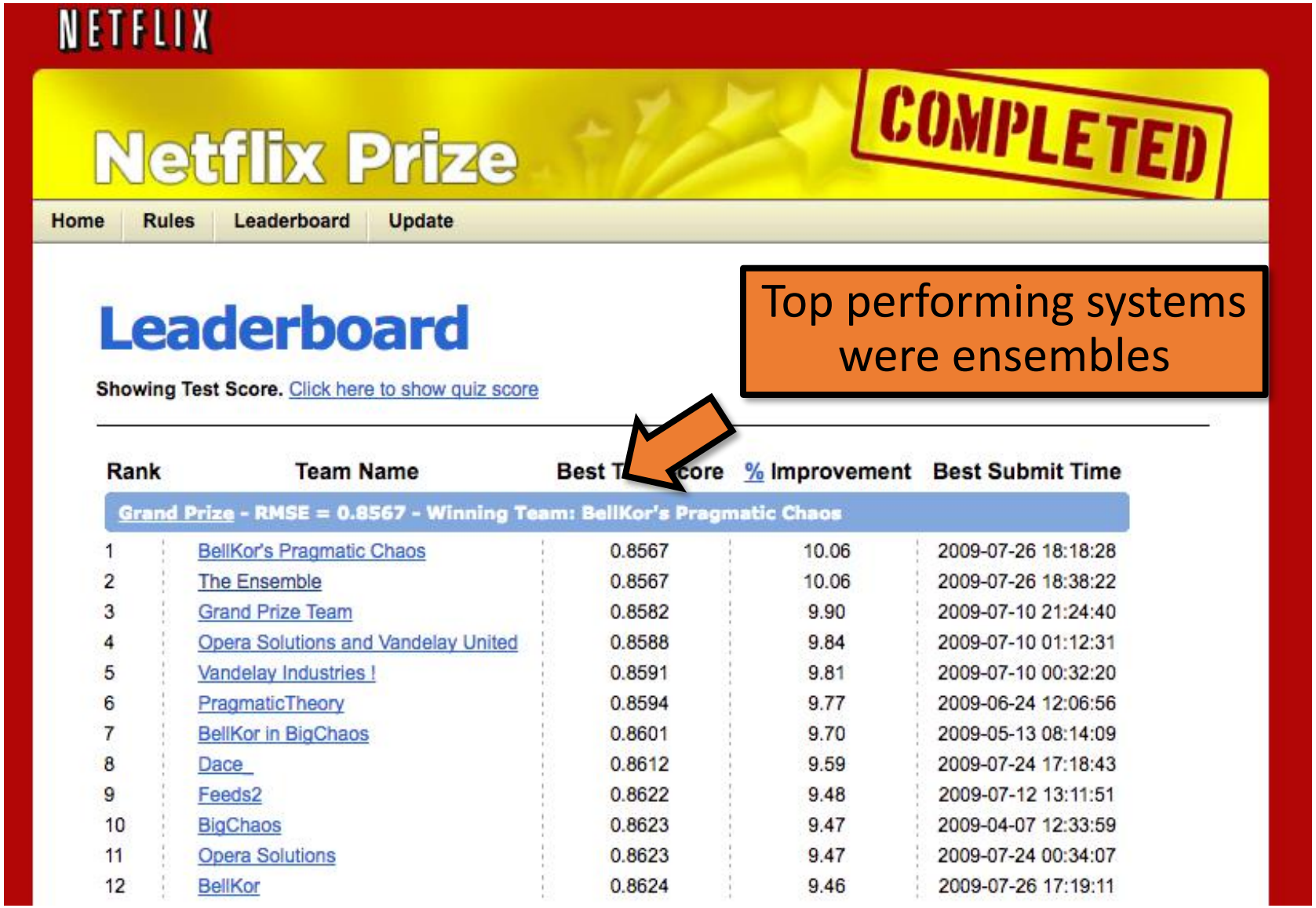
B. Why not?

Introduction to Machine Learning

Ensemble Methods

Instructor: Pat Virtue

Recommender Systems

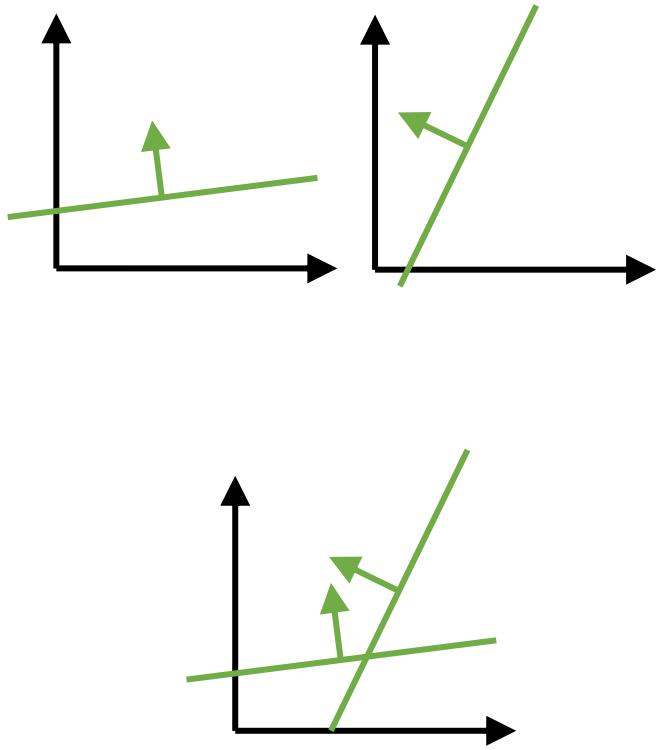


Slide credit: CMU MLD Matt Gormley

Ensemble Methods

Given: pool *A* of learners (that you know nothing about)

Goal: design a new learner that uses the predictions of the pool to make new predictions



Techniques

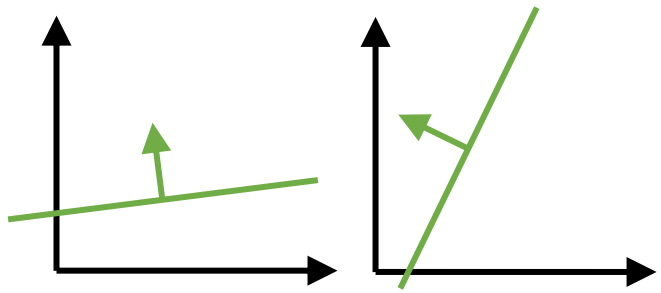
- Bagging (Bootstraping)
  - e.g. Random Forests
- Boosting
  - e.g. Adaboost

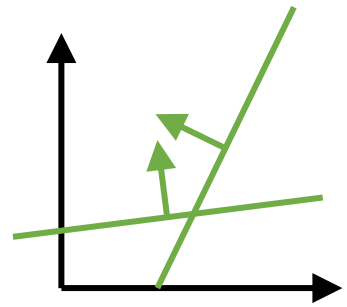
Bagging

[Breiman, 1996]

1. Run independent weak learners on bootstrap replicates (sample with replacement) of the training set

2. Average/vote over weak hypotheses





Bagging

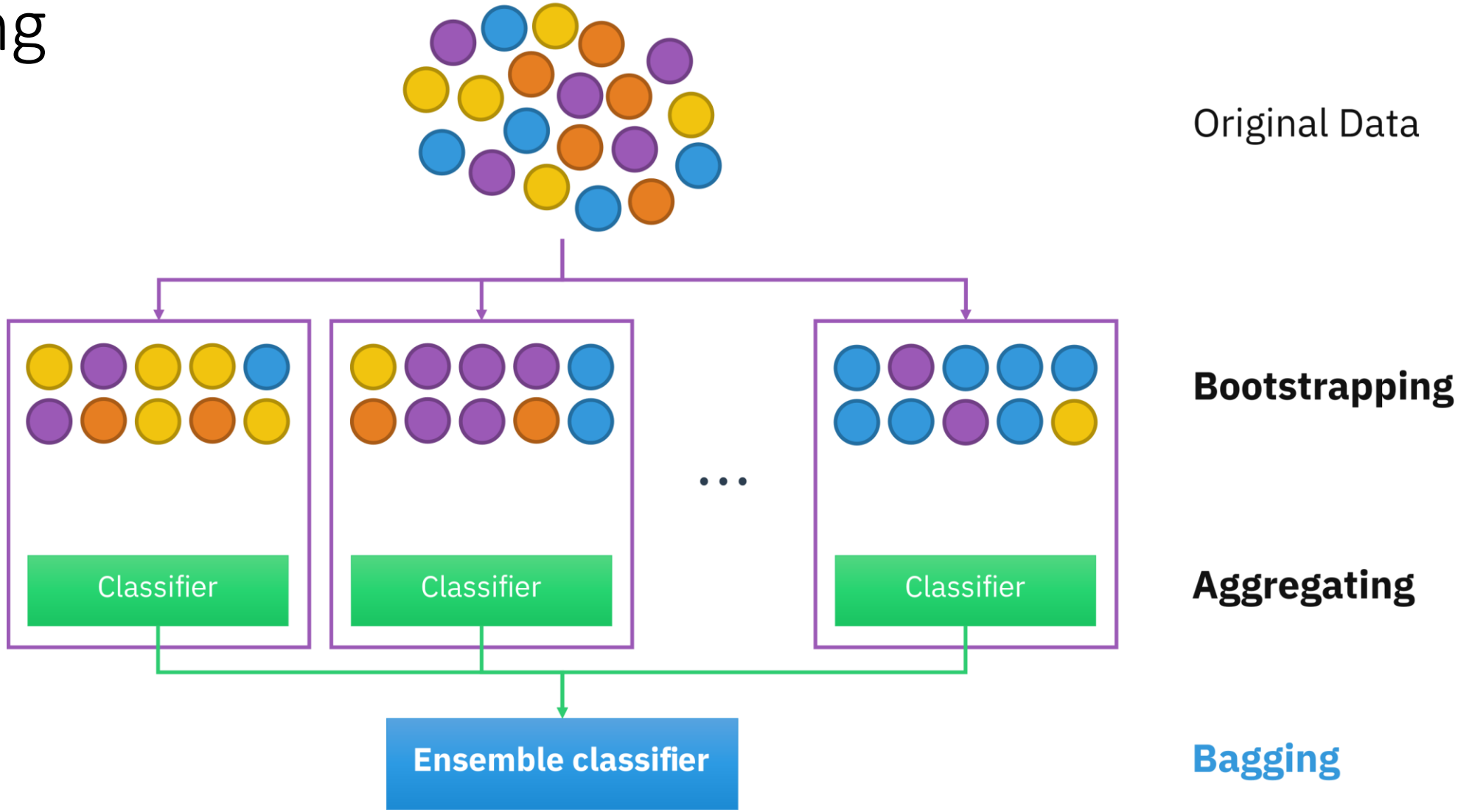
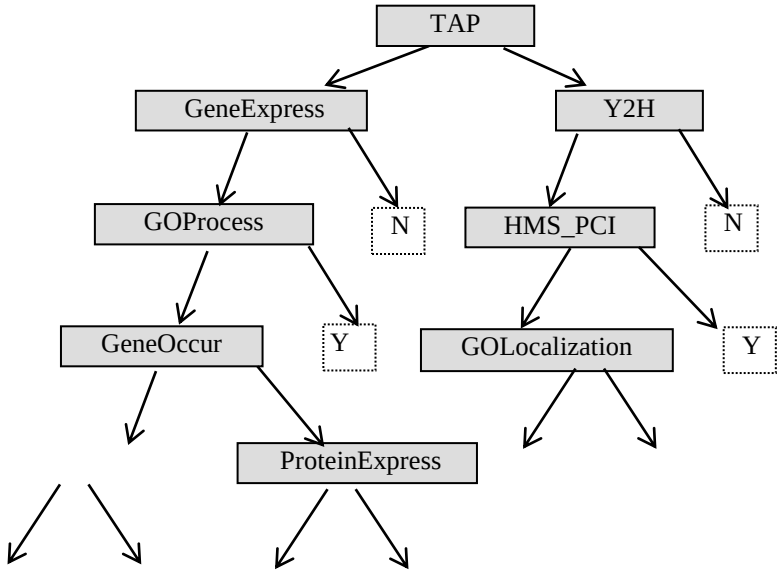


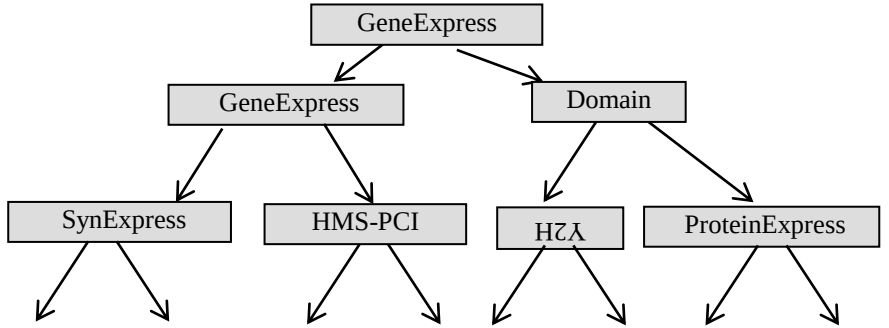
Image: https://en.m.wikipedia.org/wiki/Bootstrap_aggregating

Random Forests

Bagging over Features

- A collection of decision trees
- For each tree we select a subset of the attributes (recommended square root of |A|) and build tree using just these attributes
- An input sample is classified using majority voting





Slide credit: CMU MLD Ameet Talwalkar

Weighted Majority Algorithm

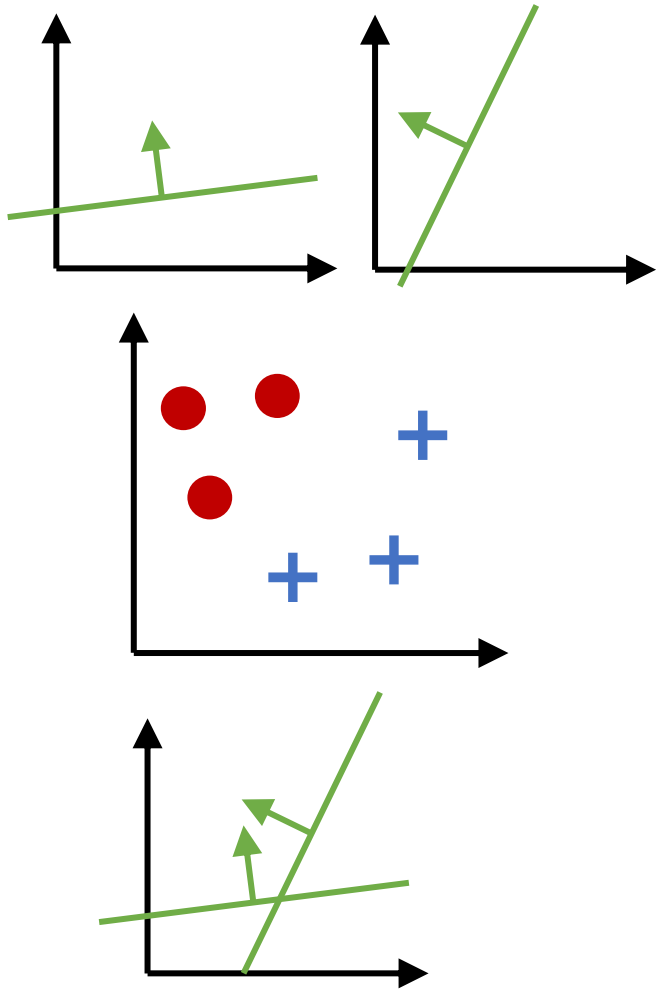
(Littlestone & Warmuth, 1994)

Given: pool *A* of binary classifiers (that you know nothing about)

Goal: design a new learner that uses the predictions of the pool to make new predictions

Algorithm:

- Initially weight all classifiers equally
- Receive a training example and predict the (weighted) majority vote of the classifiers in the pool
- Down-weight classifiers that contribute to a mistake by a factor of β



Slide credit: CMU MLD Matt Gormley

Weighted Majority Algorithm

(Littlestone & Warmuth, 1994)

Suppose we have a pool of T binary classifiers $\mathcal{A} = \{h_1, \ldots, h_T\}$ where $h_t : \mathbb{R}^M \to \{+1, -1\}$. Let α_t be the weight for classifier h_t.

Algorithm 1 Weighted Majority Algorithm

procedure WEIGHTEDMAJORITY($\mathcal{A}$, β)
  Initialize classifier weights $\alpha_t = 1, \ \forall t \in \{1, \ldots, T\}$
  for each training example $(\mathbf{x}, y)$ **do**
    Predict majority vote class (splitting ties randomly)

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

    if a mistake is made $\hat{h}(x) \neq y$ **then**
      for each classifier $t \in \{1, \ldots, T\}$ **do**
        If $h_t(x) \neq y$, then $\alpha_t \leftarrow \beta \alpha_t$

Adaboost

Comparison

Weighted Majority Algorithm

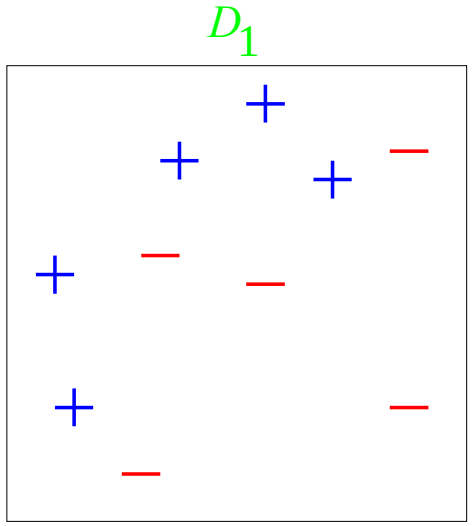
- an example of an ensemble method
- assumes the classifiers are learned ahead of time
- only learns (majority vote) weight for each classifiers

AdaBoost

- an example of a boosting method
- simultaneously learns:
  - the classifiers themselves
  - (majority vote) weight for each classifiers

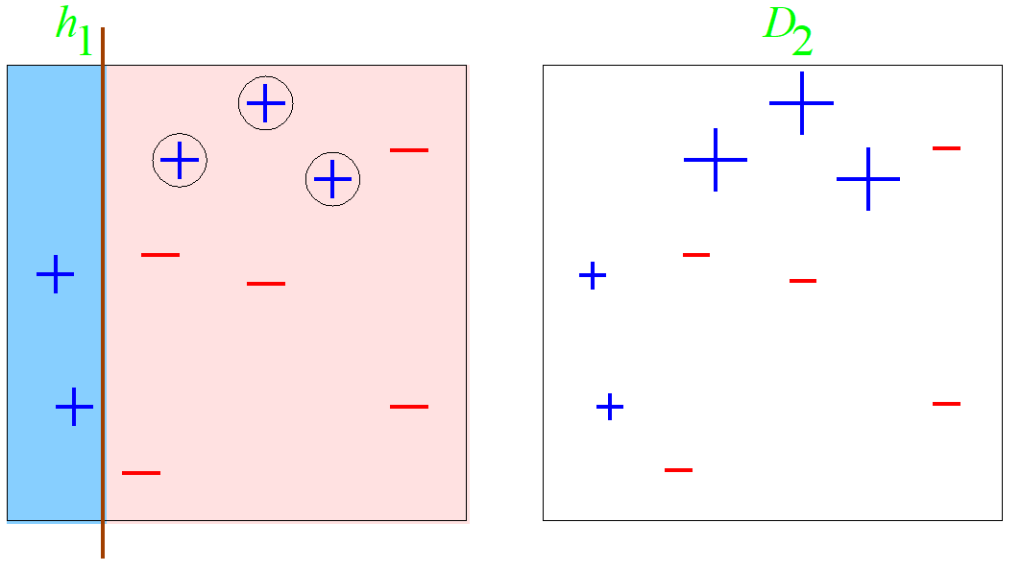
Slide credit: CMU MLD Matt Gormley

AdaBoost: Toy Example



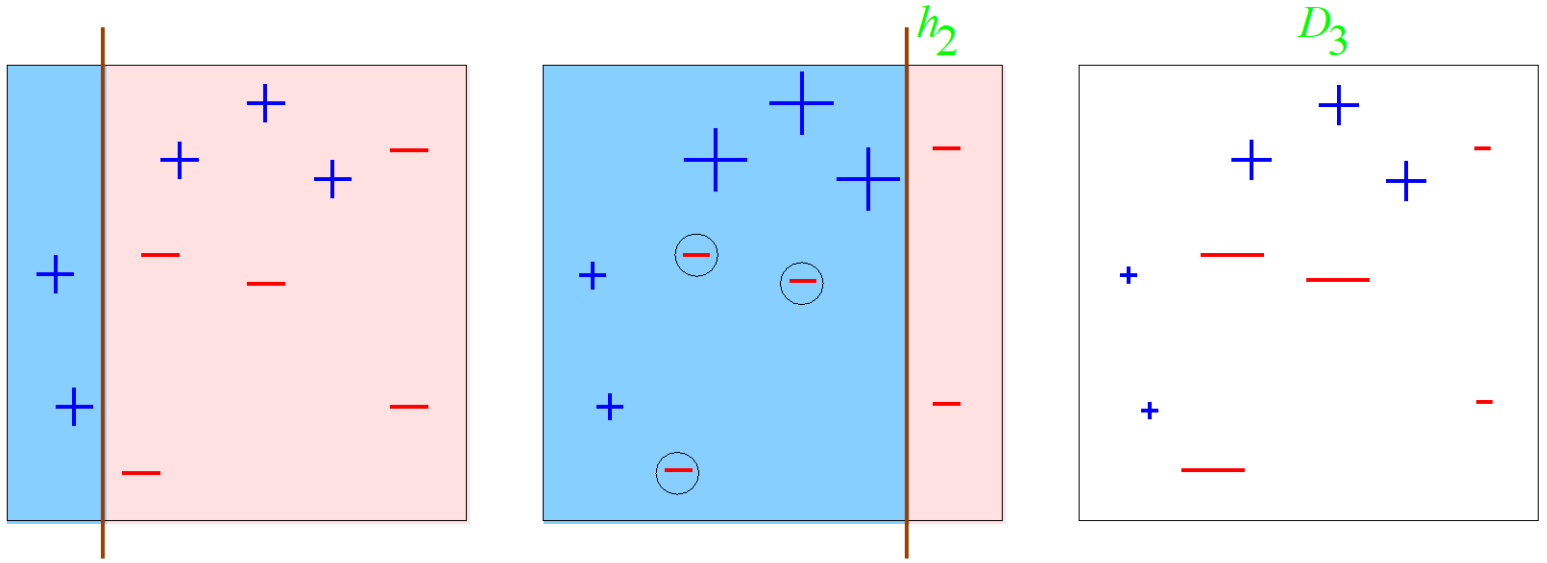
weak classifiers = vertical or horizontal half-planes

AdaBoost: Toy Example



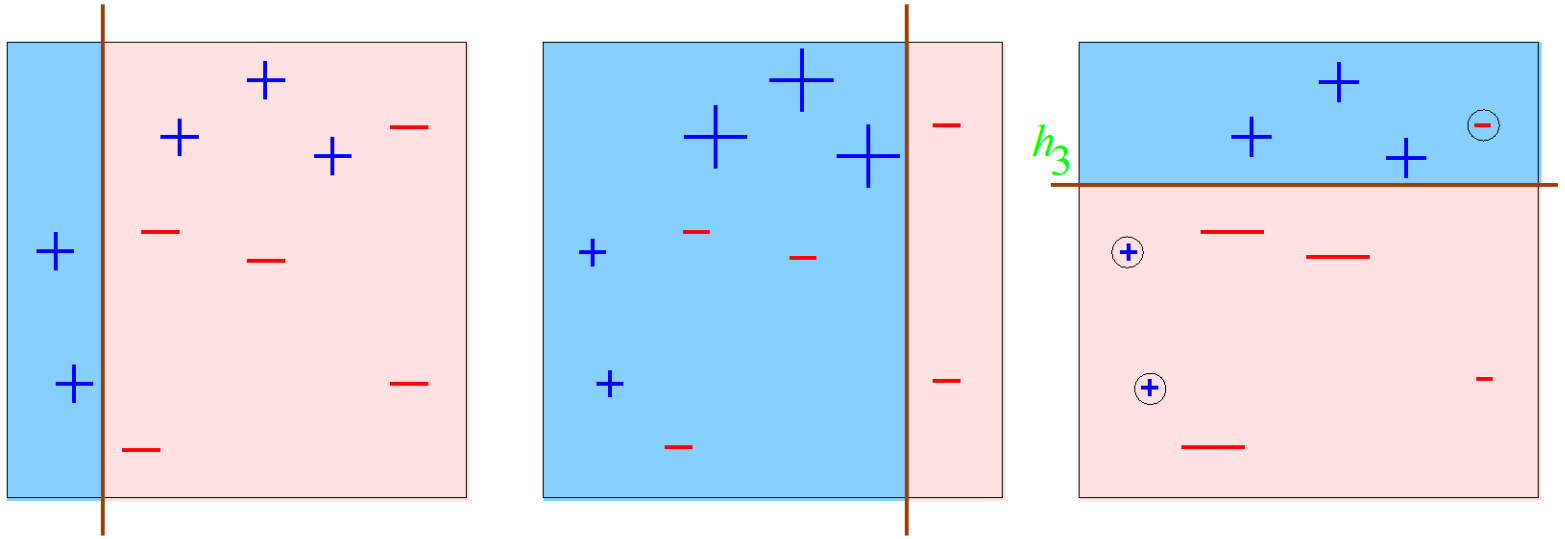
Slide from Schapire NIPS Tutorial

AdaBoost: Toy Example



Slide from Schapire NIPS Tutorial

AdaBoost: Toy Example

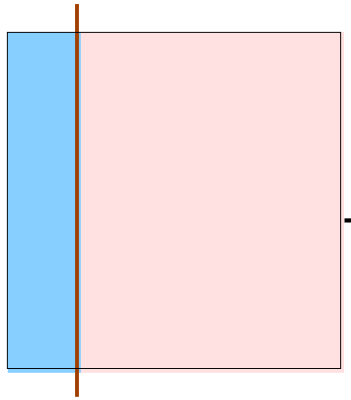
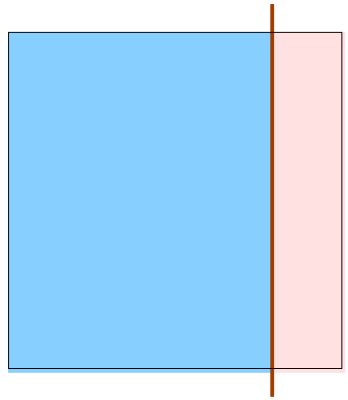
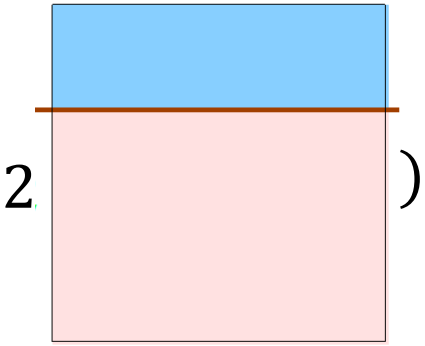


$\varepsilon_3 = 0.14$

$\alpha_3 = 0.92$

Slide from Schapire NIPS Tutorial

Piazza Poll 2-4

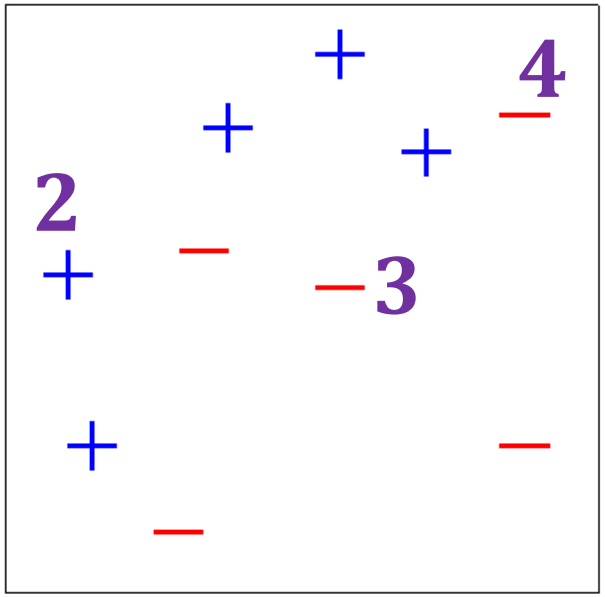
$$H_{final} = sign(\ 0.42$$  $$+0.65$$  $$+0.92$$  $$)$$

What is the final classification of points 2, 3, 4?

A.

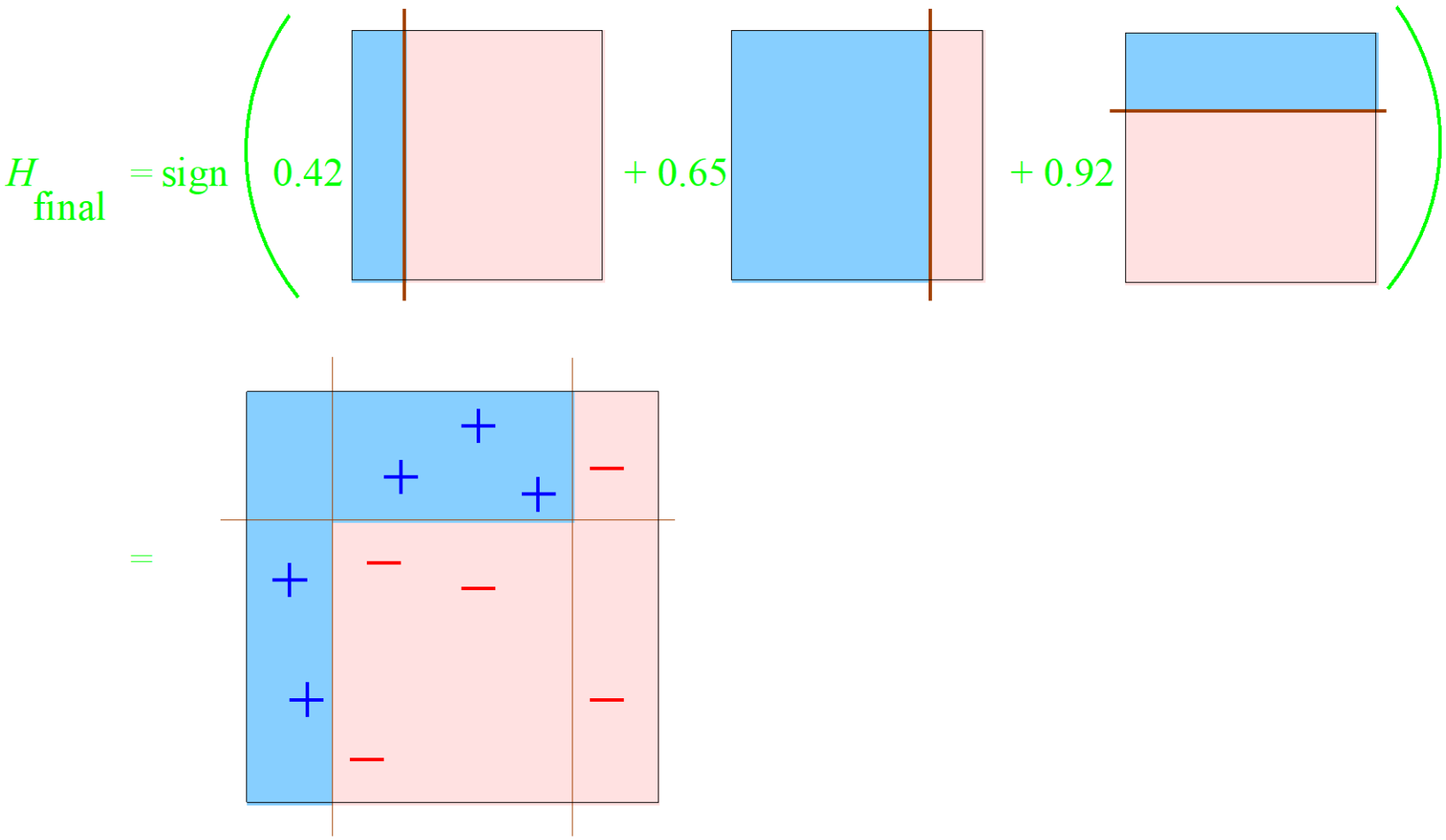
B.

C.



Slide from Schapire NIPS Tutorial

AdaBoost: Toy Example



Slide from Schapire NIPS Tutorial

AdaBoost

Given: $(x_1, y_1), \ldots, (x_m, y_m)$ where $x_i \in X$, $y_i \in Y = \{-1, +1\}$
Initialize $D_1(i) = 1/m$.
For $t = 1, \ldots, T$:

- Train weak learner using distribution D_t.
- Get weak hypothesis $h_t : X \to \{-1, +1\}$ with error

$$\epsilon_t = \Pr_{i \sim D_t} \left[h_t(x_i) \neq y_i \right].$$

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

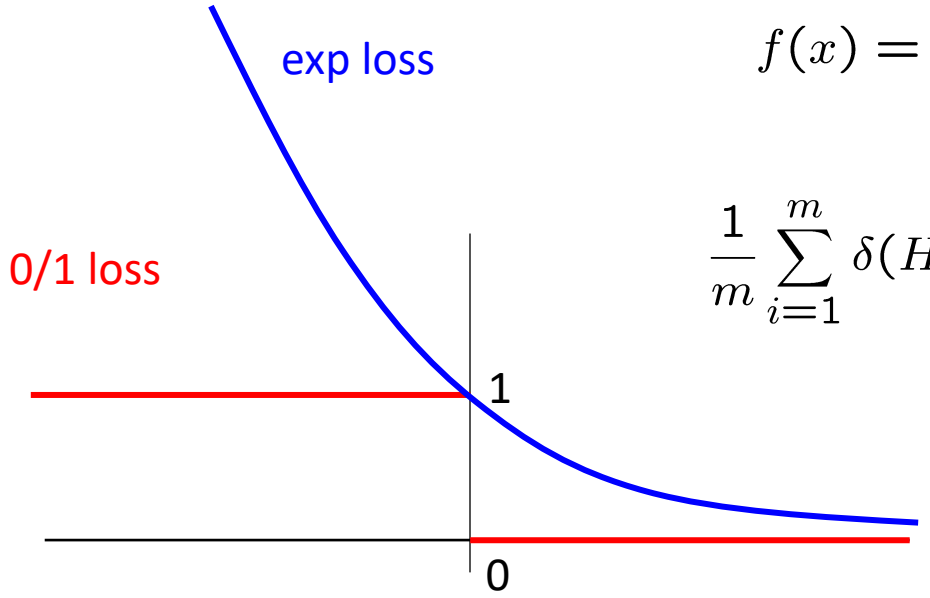
Output the final hypothesis:

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

Algorithm from Freund & Schapire, 1999

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)



$$f(x) = \sum_t \alpha_t h_t(x); H(x) = sign(f(x))$$

$$\frac{1}{m}\sum_{i=1}^{m} \delta(H(x_i) \neq y_i) \leq \frac{1}{m}\sum_{i=1}^{m} \exp(-y_i f(x_i))$$

0/1 loss exp loss

Analysis from Freund & Schapire, 1999

Analysis for Boosting

Analysis reveals:

If each weak learner h_t is slightly better than random guessing ($\varepsilon_t < 0.5$),

then training error of AdaBoost decays exponentially fast in number of rounds T.

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

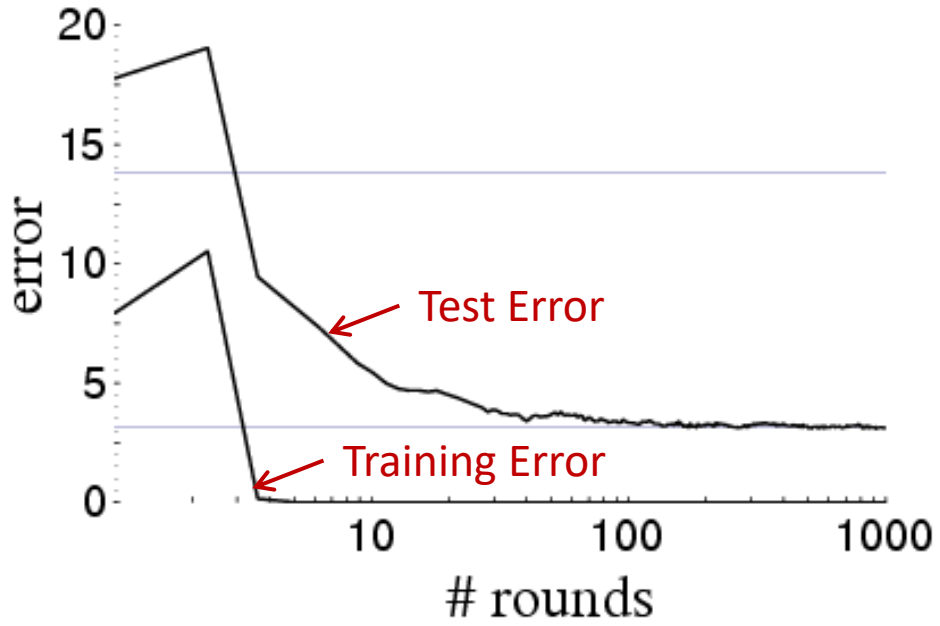
Training Error

What about test error?

Analysis from Freund & Schapire, 1999

Boosting results – Digit recognition

[Schapire, 1989]

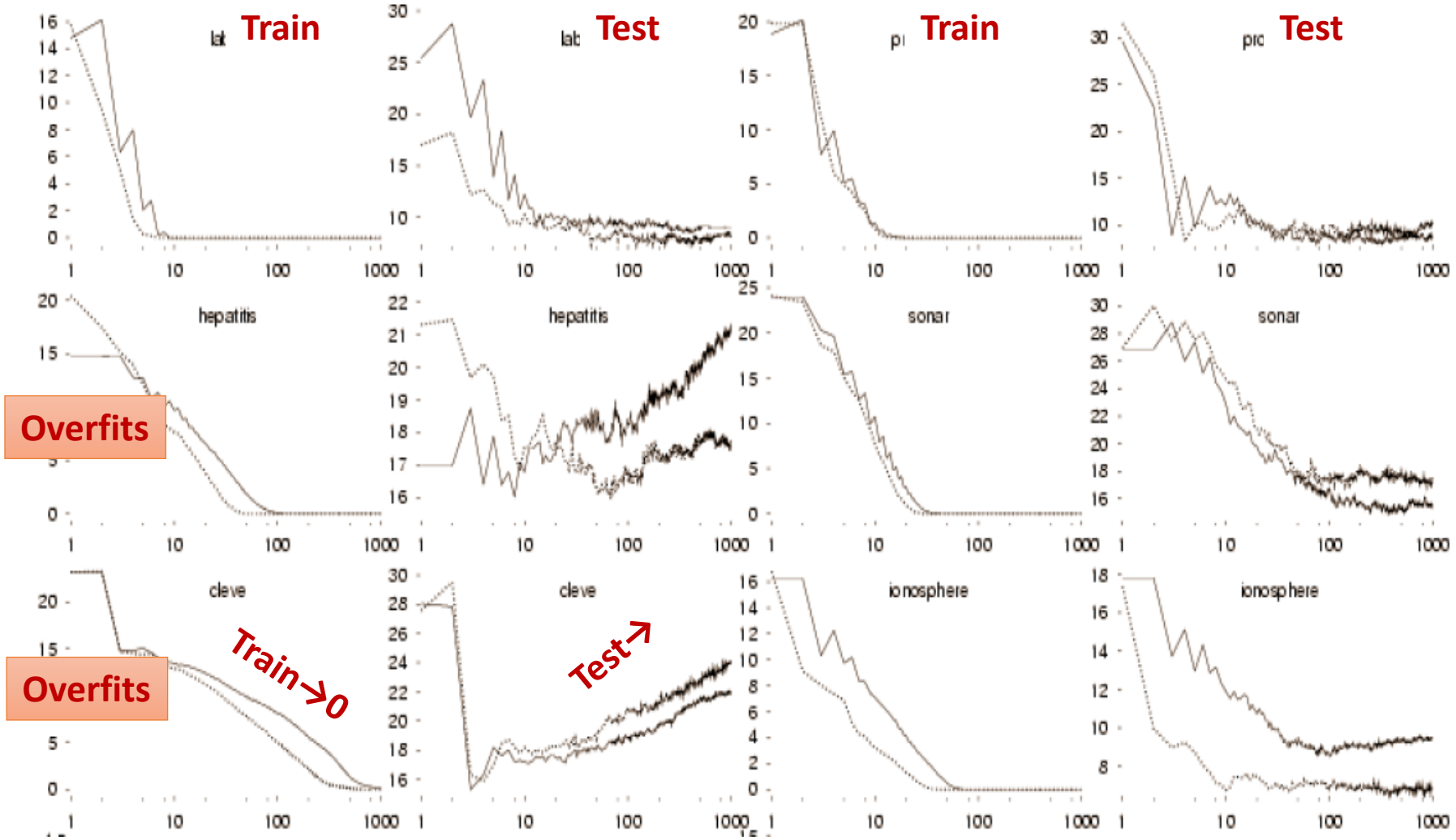


Boosting often,

- Robust to overfitting
- Test set error decreases even after training error is zero

- If margin between classes is large, 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 margin between classes is too small (high label noise) or weak learners are too complex.

AdaBoost

Given: $(x_1, y_1), \ldots, (x_m, y_m)$ where $x_i \in X$, $y_i \in Y = \{-1, +1\}$
Initialize $D_1(i) = 1/m$.
For $t = 1, \ldots, T$:

- Train weak learner using distribution D_t.
- Get weak hypothesis $h_t : X \to \{-1, +1\}$ with error

$$\epsilon_t = \Pr_{i \sim D_t} \left[h_t(x_i) \neq y_i \right].$$

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

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

Algorithm from Freund & Schapire, 1999

Dumb classifiers made Smart

Training error of final classifier is bounded by:

$$\frac{1}{m}\sum_{i=1}^{m}\delta(H(x_i)\neq y_i) \leq \prod_t Z_t = \prod_t \sqrt{1-(1-2\epsilon_t)^2}$$

$$\leq \exp\left(-2\sum_{t=1}^{T}(1/2-\epsilon_t)^2\right)$$

grows as ε_t moves away from 1/2

If each classifier is (at least slightly) better than random $\varepsilon_t < 0.5$

AdaBoost will achieve zero *training error* exponentially fast (in number of rounds T) !!

What about test error?

Generalization Error Bounds

[Freund & Schapire'95]

$$error_{true}(H) \quad \leq \quad error_{train}(H) + \tilde{\mathcal{O}}\left(\sqrt{\frac{Td}{m}}\right)$$

With high probability

Generalization Error Bounds

[Freund & Schapire'95]

$$error_{true}(H) \leq error_{train}(H) + \tilde{\mathcal{O}}\left(\sqrt{\frac{Td}{m}}\right)$$

With high probability

Boosting can overfit if T is large

Boosting often, **Contradicts experimental results**

- Robust to overfitting
- Test set error decreases even after training error is zero

Need better analysis tools – margin based bounds

Slide credit: CMU MLD Aarti Singh and Carlos Guestrin

Margin Based Bounds

[Schapire, Freund, Bartlett, Lee'98]

$$error_{true}(H) \leq \hat{\Pr}\left[\text{margin}_f(x,y) \leq \theta\right] + \tilde{O}\left(\sqrt{\frac{d}{m\theta^2}}\right)$$

With high probability

Boosting increases the margin very aggressively since it concentrates on the hardest examples.

If margin is large, more weak learners agree and hence more rounds does not necessarily imply that final classifier is getting more complex.

Bound is independent of number of rounds T!

Boosting can still overfit if margin is too small (can perform arbitrarily close to random guessing) or weak learners are too complex

Slide credit: CMU MLD Aarti Singh and Carlos Guestrin

Boosting and Logistic Regression

Logistic regression assumes:

$$P(Y = 1|X) = \frac{1}{1 + \exp(f(x))} \qquad f(x) = w_0 + \sum_j w_j x_j$$

And tries to maximize data likelihood:

$$P(\mathcal{D}|f) \overset{\text{iid}}{=} \prod_{i=1}^{m} \frac{1}{1 + \exp(-y_i f(x_i))}$$

Equivalent to minimizing log loss

$$-\log P(\mathcal{D}|f) = \sum_{i=1}^{m} \ln(1 + \exp(-y_i f(x_i)))$$

Slide credit: CMU MLD Aarti Singh and Carlos Guestrin

Boosting and Logistic Regression

Logistic regression equivalent to minimizing log loss

$$\sum_{i=1}^{m} \ln(1 + \exp(-y_i f(x_i)))$$

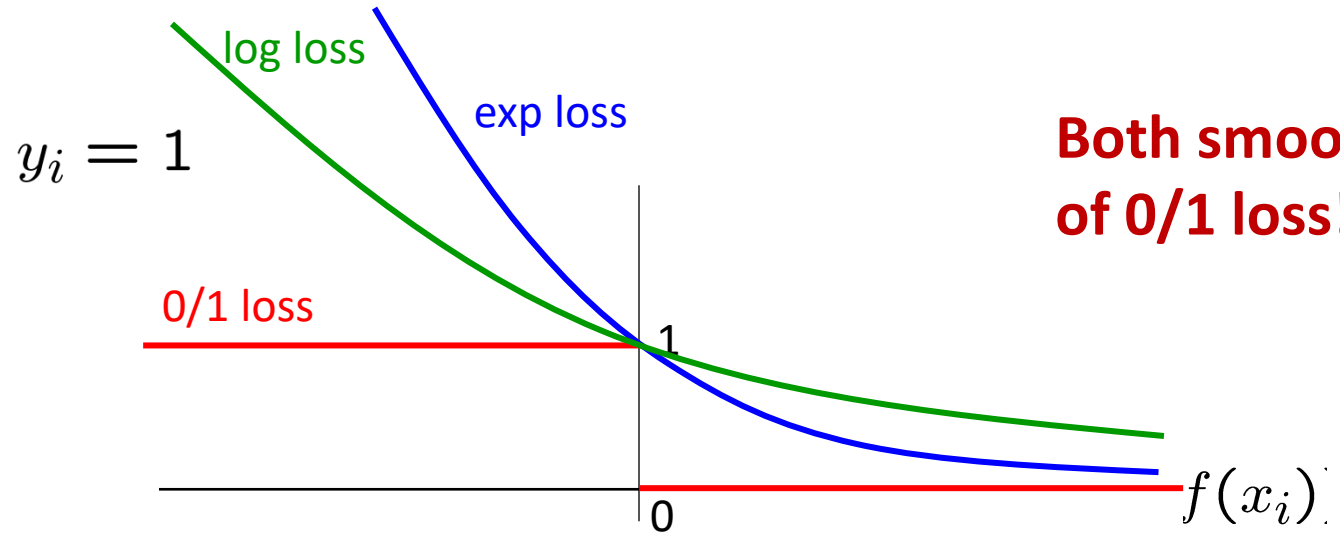
$$f(x) = w_0 + \sum_j w_j x_j$$

Boosting minimizes similar loss function!!

$$\frac{1}{m} \sum_{i=1}^{m} \exp(-y_i f(x_i))$$

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

Weighted average of weak learners



Both smooth approximations of 0/1 loss!

Analysis from: Schapire. *MSRI Workshop on Nonlinear Estimation and Classification, 2002.*

Boosting and Logistic Regression

Logistic regression:

- Minimize log loss

$$\sum_{i=1}^{m} \ln(1 + \exp(-y_i f(x_i)))$$

- Define

$$f(x) = \sum_j w_j x_j$$

where x_j predefined features

(linear classifier)

- Jointly optimize over all weights $w_0, w_1, w_2 \ldots$

Boosting:

- Minimize exp loss

$$\sum_{i=1}^{m} \exp(-y_i f(x_i))$$

- Define

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

where $h_t(x)$ defined dynamically to fit data

(not a linear classifier)

- Weights α_t learned per iteration t incrementally

Slide credit: CMU MLD Aarti Singh and Carlos Guestrin

Hard & Soft Decision

Weighted average of weak learners $f(x) = \sum_t \alpha_t h_t(x)$

Hard Decision/Predicted label: $H(x) = sign(f(x))$

Soft Decision:
(based on analogy with logistic regression)

$$P(Y = 1|X) = \frac{1}{1 + \exp(f(x))}$$

Slide credit: CMU MLD Aarti Singh and Carlos Guestrin

Bagging vs Boosting

Bagging	vs.	Boosting
Resamples data points		Reweights data points (modifies their distribution)
Weight of each classifier is the same		Weight is dependent on classifier's accuracy
Only variance reduction		Both bias and variance reduced – learning rule becomes more complex with iterations

Slide credit: CMU MLD Aarti Singh and Carlos Guestrin

Thanks to all of you!

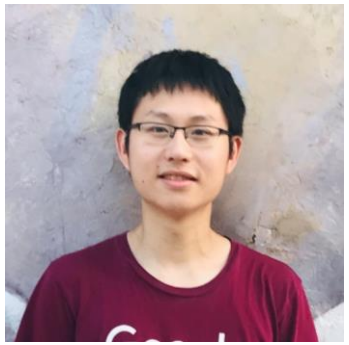
Thanks to Our Course Staff!

Teaching Assistants

Alex Singh

Annie Hu

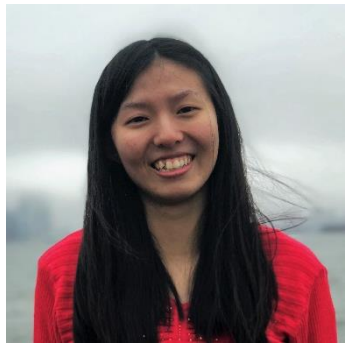
George Brown



Haoran Fei

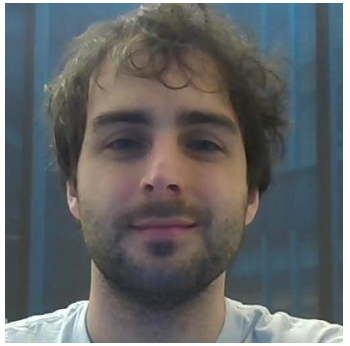
Michell Ma

Nidhi Jain



Vicky Zeng

Education Associate



Daniel Bird

Additional Analysis

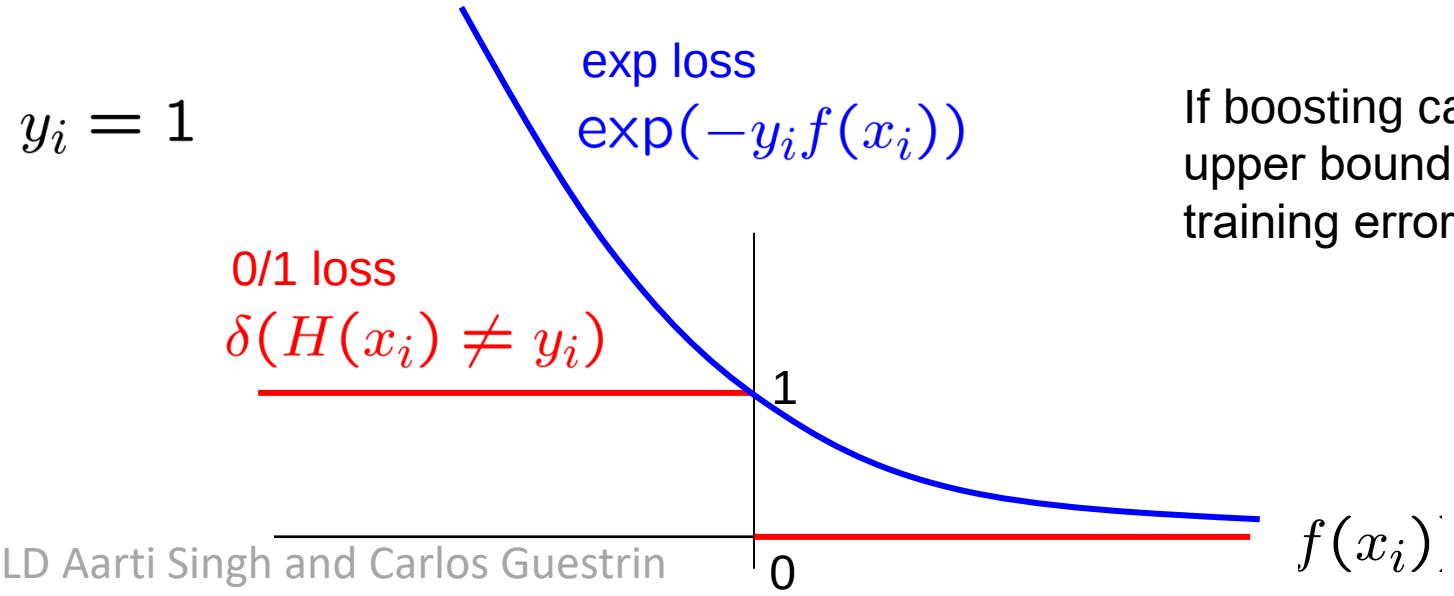
Analyzing training error

Training error of final classifier is bounded by:

$$\frac{1}{m}\sum_{i=1}^{m}\delta(H(x_i)\neq y_i) \leq \frac{1}{m}\sum_{i=1}^{m}\exp(-y_i f(x_i))$$

Convex upper bound

Where $f(x)=\sum_t \alpha_t h_t(x); H(x)=sign(f(x))$



If boosting can make upper bound → 0, then training error → 0

Analyzing training error

Analysis reveals:

- What α_t to choose for hypothesis h_t?
  What h_t to choose?

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

ε_t - weighted training error

- If each weak learner h_t is slightly better than random guessing ($\varepsilon_t < 0.5$), 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

Slide credit: CMU MLD Aarti Singh and Carlos Guestrin

Analyzing training error

Training error of final classifier is bounded by:

$$\frac{1}{m}\sum_{i=1}^{m}\delta(H(x_i)\neq y_i) \leq \frac{1}{m}\sum_{i=1}^{m}\exp(-y_i f(x_i)) = \prod_t Z_t$$

Where $f(x) = \sum_t \alpha_t h_t(x); H(x) = sign(f(x))$

Proof: **Using Weight Update Rule**

$$D_1(i) = 1/m$$

$$D_2(i) = \frac{1}{m}\frac{e^{-\alpha_1 y_i h_1(x_i)}}{Z_1}$$

$$D_3(i) = \frac{1}{m}\frac{e^{-\alpha_1 y_i h_1(x_i)} e^{-\alpha_2 y_i h_2(x_i)}}{Z_1 Z_2}$$

...

$$D_{T+1}(i) = \frac{1}{m}\frac{\exp(-y_i f(x_i))}{\prod_t Z_t}$$

Wts of all pts add to 1

$$\sum_{i=1}^{m} D_{T+1}(i) = 1$$

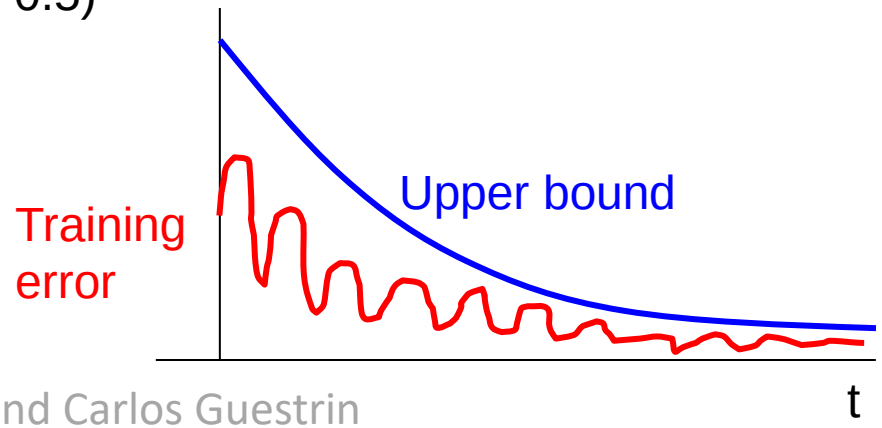
Analyzing training error

Training error of final classifier is bounded by:

$$\frac{1}{m}\sum_{i=1}^{m}\delta(H(x_i)\neq y_i) \leq \frac{1}{m}\sum_{i=1}^{m}\exp(-y_i f(x_i)) = \prod_t Z_t$$

Where $f(x) = \sum_t \alpha_t h_t(x); H(x) = sign(f(x))$

If $Z_t < 1$, training error decreases exponentially (even though weak learners may not be good ε_t ~0.5)



What α_t to choose for hypothesis h_t?

Training error of final classifier is bounded by:

$$\frac{1}{m}\sum_{i=1}^{m}\delta(H(x_i)\neq y_i) \leq \frac{1}{m}\sum_{i=1}^{m}\exp(-y_i f(x_i)) = \prod_t Z_t$$

Where $f(x) = \sum_t \alpha_t h_t(x); H(x) = sign(f(x))$

If we minimize $\prod_t Z_t$, we minimize our training error

We can tighten this bound greedily, by choosing α_t and h_t on each iteration to minimize Z_t.

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

Slide credit: CMU MLD Aarti Singh and Carlos Guestrin

What α_t to choose for hypothesis h_t?

We can minimize this bound by choosing α_t on each iteration to minimize Z_t.

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

For boolean target function, this is accomplished by [Freund & Schapire '97]:

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

Proof:

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

$$\frac{\partial Z_t}{\alpha_t} = \epsilon_t e^{\alpha_t} - (1-\epsilon_t) e^{-\alpha_t} = 0 \qquad \Rightarrow e^{2\alpha_t} = \frac{1-\epsilon_t}{\epsilon_t}$$

Slide credit: CMU MLD Aarti Singh and Carlos Guestrin

Bounding the error with choice of α_t

We can minimize this bound by choosing α_t on each iteration to minimize Z_t.

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

For boolean target function, this is accomplished by [Freund & Schapire '97]:

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

Proof:

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

Slide credit: CMU MLD Aarti Singh and Carlos Guestrin

Dumb classifiers made Smart

Training error of final classifier is bounded by:

$$\frac{1}{m}\sum_{i=1}^{m}\delta(H(x_i)\neq y_i) \leq \prod_t Z_t = \prod_t \sqrt{1-(1-2\epsilon_t)^2}$$

$$\leq \exp\left(-2\sum_{t=1}^{T}(1/2-\epsilon_t)^2\right)$$

grows as ε_t moves away from 1/2

If each classifier is (at least slightly) better than random $\varepsilon_t < 0.5$

AdaBoost will achieve zero *training error* exponentially fast (in number of rounds T) !!

What about test error?

Slide credit: CMU MLD Aarti Singh and Carlos Guestrin

Generalization Error Bounds

[Freund & Schapire'95]

$$error_{true}(H) \leq error_{train}(H) + \tilde{\mathcal{O}}\left(\sqrt{\frac{Td}{m}}\right)$$

With high probability

Boosting can overfit if T is large

Boosting often, **Contradicts experimental results**

- Robust to overfitting
- Test set error decreases even after training error is zero

Need better analysis tools – margin based bounds

Margin Based Bounds

[Schapire, Freund, Bartlett, Lee'98]

$$error_{true}(H) \leq \hat{\Pr}\left[\text{margin}_f(x,y) \leq \theta\right] + \tilde{O}\left(\sqrt{\frac{d}{m\theta^2}}\right)$$

With high probability

Boosting increases the margin very aggressively since it concentrates on the hardest examples.

If margin is large, more weak learners agree and hence more rounds does not necessarily imply that final classifier is getting more complex.

Bound is independent of number of rounds T!

Boosting can still overfit if margin is too small (can perform arbitrarily close to random guessing) or weak learners are too complex

Slide credit: CMU MLD Aarti Singh and Carlos Guestrin