10-701: Introduction to Machine Learning

Lecture 22 –Boosting

Hoda Heidari

* Slides adopted from F24 offering of 10701 by Henry Chai.

Bias-Variance Decomposition

- Suppose y=f(x)+ε, where ε is noise with
$$\mathbb{E}[\varepsilon] = 0, Var(\varepsilon) = \sigma^2.$$
- The expected squared prediction error at a point x:

$$\mathbb{E}_{D,\varepsilon}\left[\left(y - \hat{f}(x)\right)^2\right] =$$
$$= \underbrace{\left(f(x) - \mathbb{E}\left[\hat{f}(x)\right]\right)^2}_{Bias^2} + \underbrace{\mathbb{E}\left[\left(\hat{f}(x) - \mathbb{E}\left[\hat{f}(x)\right]\right)^2\right]}_{Variance} + \sigma^2$$

Ensemble Learning

- **Key idea:** Different models make different errors. By aggregating their predictions, ensembles can reduce variance, bias, or both—leading to better accuracy and robustness than any single model alone.
- **Common ensemble methods**
  - **Bagging (Bootstrap Aggregating):**
    Trains many models independently on different random subsets of the data to reduce <u>variance and overfitting</u>.
  - **Boosting:**
    Trains models sequentially, with each new model focusing on the errors of the previous one to reduce <u>bias</u>.
  - **Stacking:**
    Combines predictions from multiple models using another model (a *meta-learner*) that learns how best to blend them.

Decision Trees: Pros & Cons

- Pros
  - Interpretable
  - Efficient (computational cost and storage)
  - …
- Cons
  - Learned greedily: each split only considers the immediate impact on the splitting criterion
  - Prone to overfit
  - High variance
    - Can be addressed via **bagging** → random forests
  - Limited expressivity (especially short trees, i.e., stumps)
    - Can be addressed via **boosting**

AdaBoost

- Intuition: iteratively reweight inputs, giving more weight to inputs that are difficult-to-predict correctly
- Analogy:
  - You all have to take a test (😱) …
  - … but you're going to be taking it one at a time.
  - After you finish, you get to tell the next person the questions you struggled with.
  - Hopefully, they can cover for you because…
  - … if "enough" of you get a question right, you'll all receive full credit for that problem

AdaBoost

- Input: $\mathcal{D}\left(y^{(n)} \in \{-1, +1\}\right), T$
- Initialize data point weights: $\omega_0^{(1)}, \ldots, \omega_0^{(N)} = \frac{1}{N}$
- For $t = 1, \ldots, T$
  1. Train a weak learner, h_t, by minimizing the *weighted* training error
  2. Compute the *weighted* training error of h_t:
  $$\epsilon_t = \sum_{n=1}^{N} \omega_{t-1}^{(n)} \mathbb{1}\left(y^{(n)} \neq h_t\left(\boldsymbol{x}^{(n)}\right)\right)$$
  3. Compute the **importance** of h_t:
  $$\alpha_t = \frac{1}{2} \log\left(\frac{1-\epsilon_t}{\epsilon_t}\right)$$
  4. Update the data point weights:

$$\omega_t^{(n)} = \frac{\omega_{t-1}^{(n)}}{Z_t} \times \begin{cases} e^{-\alpha_t} \text{ if } h_t\left(\boldsymbol{x}^{(n)}\right) = y^{(n)} \\ e^{\alpha_t} \quad \text{if } h_t\left(\boldsymbol{x}^{(n)}\right) \neq y^{(n)} \end{cases} = \frac{\omega_{t-1}^{(n)} e^{-\alpha_t y^{(n)} h_t\left(\boldsymbol{x}^{(n)}\right)}}{Z_t}$$

- Output: an aggregated hypothesis

$$g_T(\boldsymbol{x}) = \operatorname{sign}\left(H_T(\boldsymbol{x})\right)$$
$$= \operatorname{sign}\left(\sum_{t=1}^{T} \alpha_t h_t(\boldsymbol{x})\right)$$

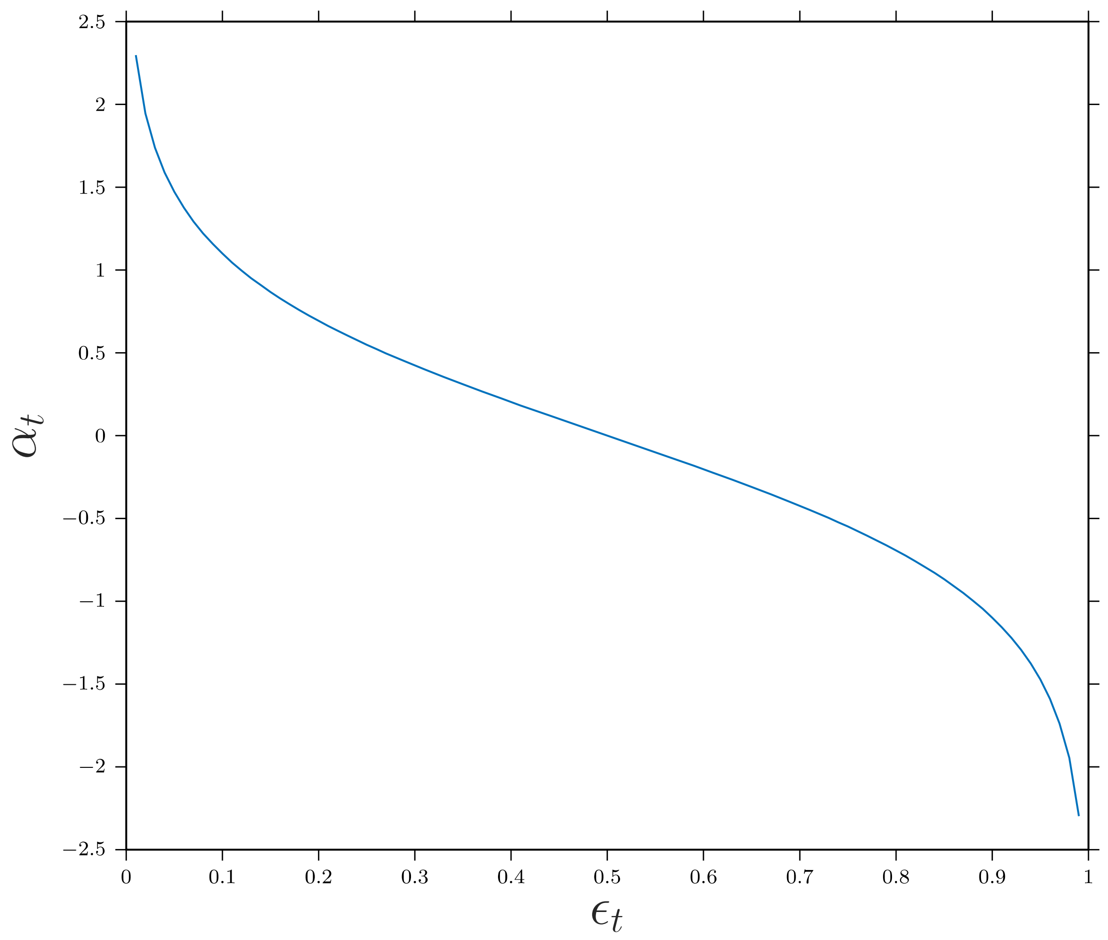
Setting α_t

α_t determines the contribution of h_t to the final, aggregated hypothesis:

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

Intuition: we want good weak learners to have high importances

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



Updating $\omega^{(n)}$

- Intuition: we want incorrectly classified inputs to receive a higher weight in the next round

$$\omega_t^{(n)} = \frac{\omega_{t-1}^{(n)}}{Z_t} \times \begin{cases} e^{-\alpha_t} \text{ if } h_t\left(\boldsymbol{x}^{(n)}\right) = y^{(n)} \\ e^{\alpha_t} \quad \text{if } h_t\left(\boldsymbol{x}^{(n)}\right) \neq y^{(n)} \end{cases}$$

- If $\epsilon_t < \frac{1}{2}$, ….

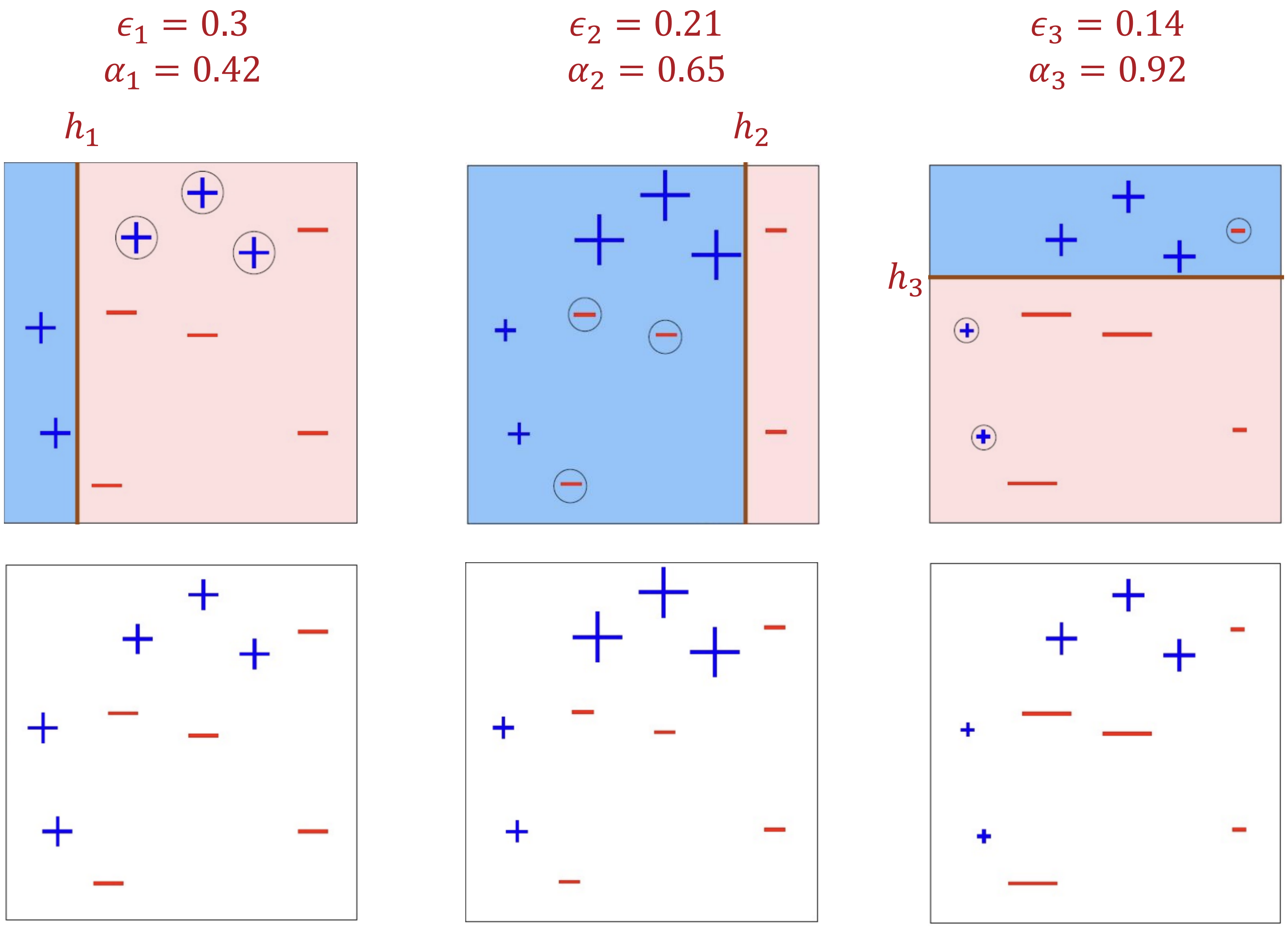
Updating $\omega^{(n)}$

- Intuition: we want incorrectly classified inputs to receive a higher weight in the next round

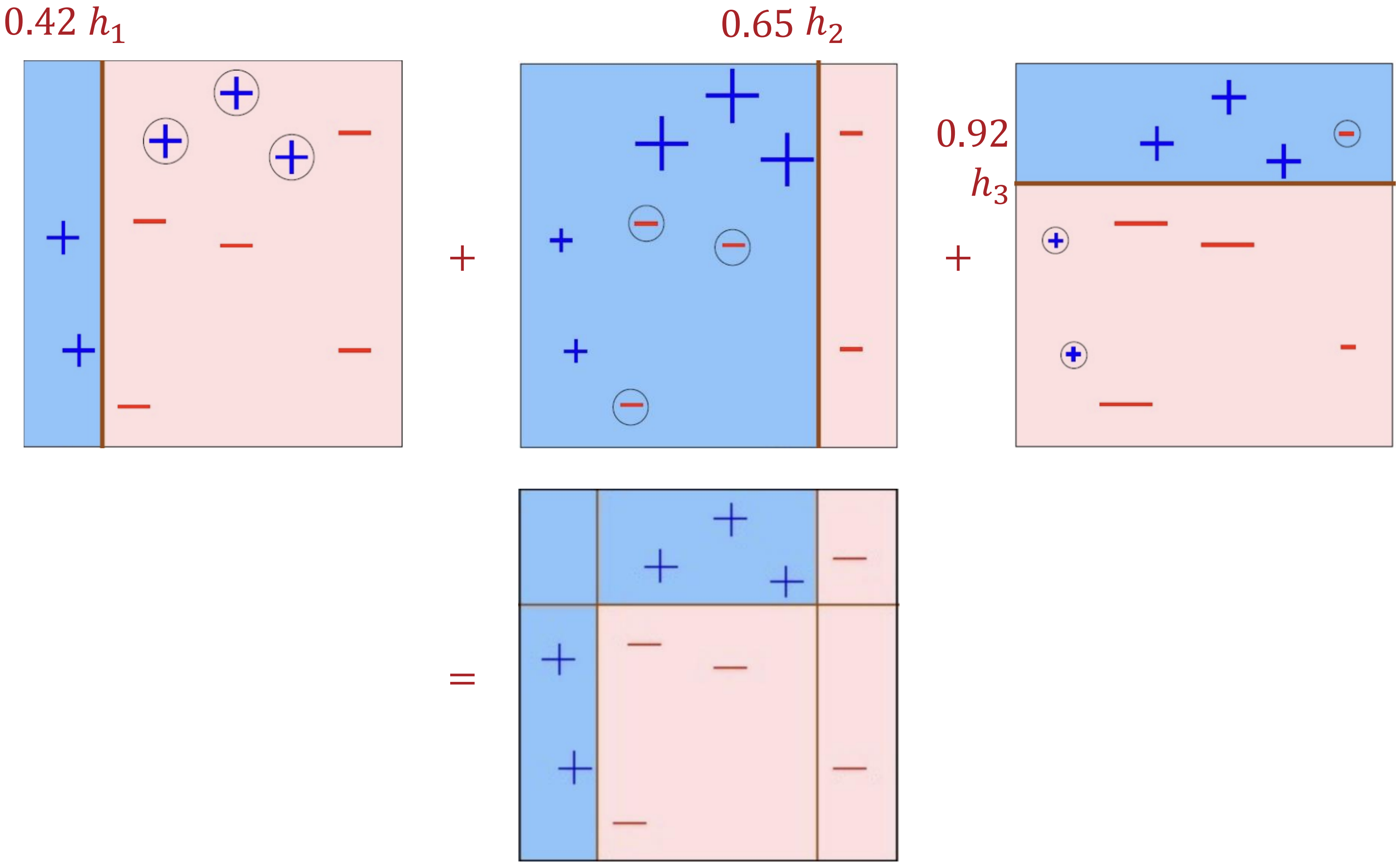
$$\omega_t^{(n)} = \frac{\omega_{t-1}^{(n)}}{Z_t} \times \begin{cases} e^{-\alpha_t} \text{ if } h_t\left(\boldsymbol{x}^{(n)}\right) = y^{(n)} \\ e^{\alpha_t} \quad \text{if } h_t\left(\boldsymbol{x}^{(n)}\right) \neq y^{(n)} \end{cases} = \frac{\omega_{t-1}^{(n)} e^{-\alpha_t y^{(n)} h_t\left(\boldsymbol{x}^{(n)}\right)}}{Z_t}$$

- If $\epsilon_t < \frac{1}{2}$, then $\frac{1-\epsilon_t}{\epsilon_t} > 1$
- If $\frac{1-\epsilon_t}{\epsilon_t} > 1$, then $\alpha_t = \frac{1}{2}\log\left(\frac{1-\epsilon_t}{\epsilon_t}\right) > 0$
- If $\alpha_t > 0$, then $e^{-\alpha_t} < 1$ and $e^{\alpha_t} > 1$

AdaBoost: Example



AdaBoost: Example



Why AdaBoost?

1. If you want to use weak learners …
2. … and want your final hypothesis to be a weighted combination of weak learners, …
3. … then Adaboost greedily minimizes the exponential loss:

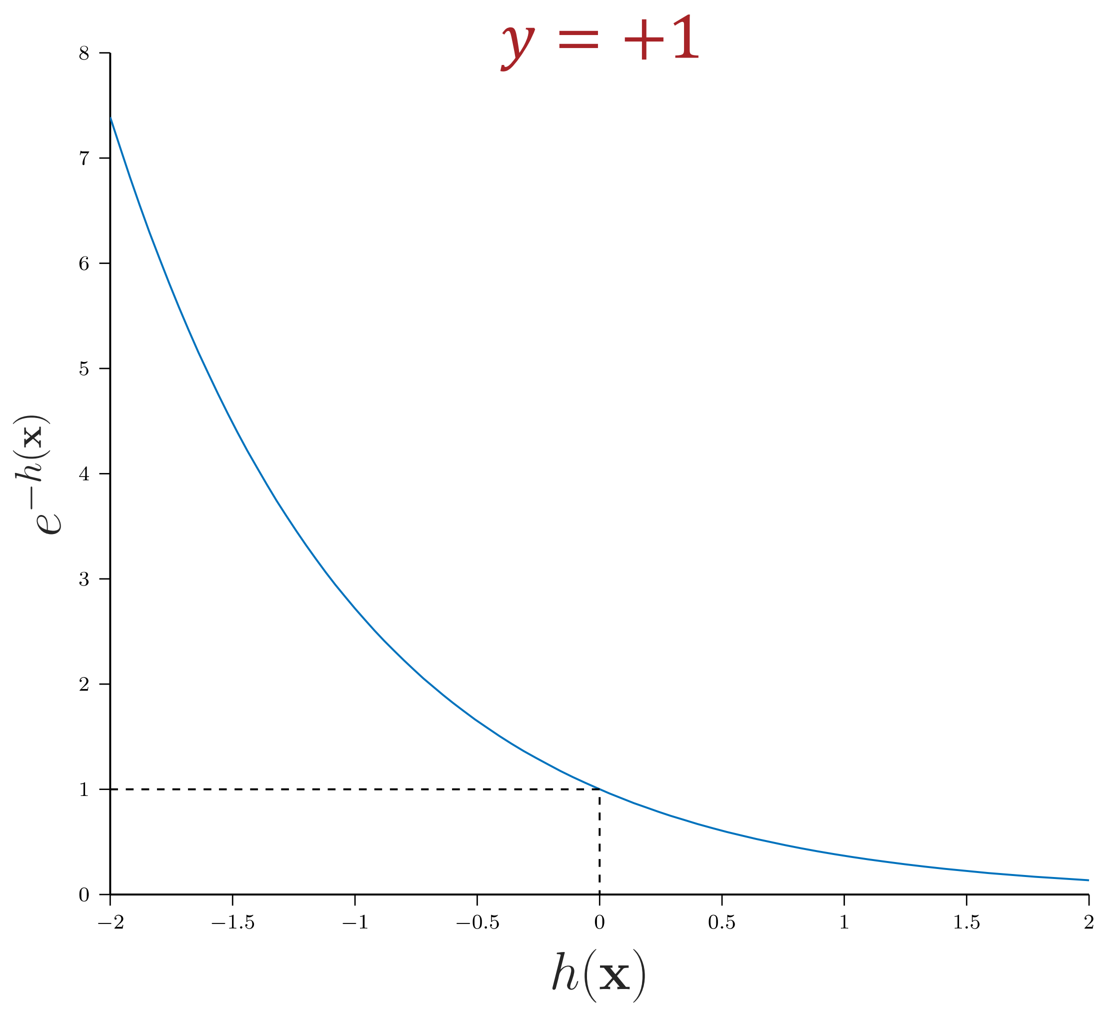
$$e(h(\boldsymbol{x}), y) = e^{(-yh(\boldsymbol{x}))}$$

1. Because they're low variance / computational constraints
2. Because weak learners are not great on their own
3. Because the exponential loss upper bounds binary error!

Exponential Loss

$$e(h(\boldsymbol{x}), y) = e^{(-yh(\boldsymbol{x}))}$$

The more $h(\boldsymbol{x})$ "agrees with" y, the smaller the loss and the more $h(\boldsymbol{x})$ "disagrees with" y, the greater the loss



Exponential Loss

- Claim:

$$\frac{1}{N}\sum_{n=1}^{N} e^{\left(-y^{(n)}h\left(\boldsymbol{x}^{(n)}\right)\right)} \geq \frac{1}{N}\sum_{n=1}^{N} \mathbb{1}\left(\text{sign}\left(h\left(\boldsymbol{x}^{(n)}\right)\right) \neq y^{(n)}\right)$$

- Consequence:

$$\frac{1}{N}\sum_{n=1}^{N} e^{\left(-y^{(n)}h\left(\boldsymbol{x}^{(n)}\right)\right)} \to 0$$

$$\Longrightarrow \frac{1}{N}\sum_{n=1}^{N} \mathbb{1}\left(\text{sign}\left(h\left(\boldsymbol{x}^{(n)}\right)\right) \neq y^{(n)}\right) \to 0$$

Exponential Loss

- Claim: if $g_T = \text{sign}(H_T)$ is the Adaboost hypothesis, then

$$\frac{1}{N}\sum_{n=1}^{N} e^{\left(-y^{(n)}H_T\left(\boldsymbol{x}^{(n)}\right)\right)} = \prod_{t=1}^{T} Z_t$$

- Proof:

Exponential Loss

- Claim: if $g_T = \text{sign}(H_T)$ is the Adaboost hypothesis, then

$$\frac{1}{N}\sum_{n=1}^{N} e^{\left(-y^{(n)}H_T\left(\boldsymbol{x}^{(n)}\right)\right)} = \prod_{t=1}^{T} Z_t$$

- Proof:

$$\omega_0^{(n)} = \frac{1}{N}, \ \omega_1^{(n)} = \frac{e^{-\alpha_1 y^{(n)} h_1\left(\boldsymbol{x}^{(n)}\right)}}{N Z_1}, \ \omega_2^{(n)} = \frac{e^{-\alpha_1 y^{(n)} h_1\left(\boldsymbol{x}^{(n)}\right)} e^{-\alpha_2 y^{(n)} h_2\left(\boldsymbol{x}^{(n)}\right)}}{N Z_1 Z_2}$$

$$\omega_T^{(n)} = \frac{\prod_{t=1}^{T} e^{-\alpha_t y^{(n)} h_t\left(\boldsymbol{x}^{(n)}\right)}}{N \prod_{t=1}^{T} Z_t} = \frac{e^{-y^{(n)} \sum_{t=1}^{T} \alpha_t h_t\left(\boldsymbol{x}^{(n)}\right)}}{N \prod_{t=1}^{T} Z_t} = \frac{e^{-y^{(n)} H_T\left(\boldsymbol{x}^{(n)}\right)}}{N \prod_{t=1}^{T} Z_t}$$

$$\sum_{n=1}^{N} \omega_T^{(n)} = \sum_{n=1}^{N} \frac{e^{-y^{(n)} H_T\left(\boldsymbol{x}^{(n)}\right)}}{N \prod_{t=1}^{T} Z_t} = 1 \Rightarrow \frac{1}{N}\sum_{n=1}^{N} e^{-y^{(n)} H_T\left(\boldsymbol{x}^{(n)}\right)} = \prod_{t=1}^{T} Z_t \ \blacksquare$$

Exponential Loss

- Claim: if $g_T = \text{sign}(H_T)$ is the Adaboost hypothesis, then

$$\frac{1}{N}\sum_{n=1}^{N} e^{\left(-y^{(n)}H_T\left(\boldsymbol{x}^{(n)}\right)\right)} = \prod_{t=1}^{T} Z_t$$

- Consequence: one way to minimize the exponential training loss is to greedily minimize Z_t, i.e., in each iteration, make the normalization constant as small as possible by tuning α_t.

Greedy Exponential Loss Minimization

$$Z_t = \sum_{n=1}^{N} \omega_{t-1}^{(n)} e^{-(a) y^{(n)} h_t(\boldsymbol{x}^{(n)})}$$

Greedy Exponential Loss Minimization

$$Z_t = \sum_{n=1}^{N} \omega_{t-1}^{(n)} e^{-(a) y^{(n)} h_t(\boldsymbol{x}^{(n)})}$$

$$= \sum_{y^{(n)} = h_t(\boldsymbol{x}^{(n)})} \omega_{t-1}^{(n)} e^{-(a)} + \sum_{y^{(n)} \neq h_t(\boldsymbol{x}^{(n)})} \omega_{t-1}^{(n)} e^{(a)}$$

$$= e^{-(a)} \sum_{y^{(n)} = h_t(\boldsymbol{x}^{(n)})} \omega_{t-1}^{(n)} + e^{(a)} \sum_{y^{(n)} \neq h_t(\boldsymbol{x}^{(n)})} \omega_{t-1}^{(n)}$$

$$= e^{-a}(1 - \epsilon_t) + e^{a} \epsilon_t$$

Greedy Exponential Loss Minimization

$$Z_t = e^{-a}(1 - \epsilon_t) + e^{a}\epsilon_t$$

Greedy Exponential Loss Minimization

$$Z_t = e^{-a}(1 - \epsilon_t) + e^{a}\epsilon_t$$

$$\frac{\partial Z_t}{\partial a} = -e^{-a}(1 - \epsilon_t) + e^{a}\epsilon_t \Longrightarrow -e^{-\hat{a}}(1 - \epsilon_t) + e^{\hat{a}}\epsilon_t = 0$$

$$\Longrightarrow e^{\hat{a}}\epsilon_t = e^{-\hat{a}}(1 - \epsilon_t)$$

$$\Longrightarrow e^{2\hat{a}} = \frac{1 - \epsilon_t}{\epsilon_t}$$

$$\Longrightarrow \hat{a} = \frac{1}{2}\log\left(\frac{1 - \epsilon_t}{\epsilon_t}\right) = \alpha_t$$

$$\frac{\partial^2 Z_t}{\partial a^2} = e^{-a}(1 - \epsilon_t) + e^{a}\epsilon_t > 0$$

Normalizing $\omega^{(n)}$

$$Z_t = \sum_{n=1}^{N} \omega_{t-1}^{(n)} e^{-\alpha_t y^{(n)} h_t(\boldsymbol{x}^{(n)})}$$

$$= \sum_{y^{(n)} = h_t(\boldsymbol{x}^{(n)})} \omega_{t-1}^{(n)} e^{-\alpha_t} + \sum_{y^{(n)} \neq h_t(\boldsymbol{x}^{(n)})} \omega_{t-1}^{(n)} e^{\alpha_t}$$

Normalizing $\omega^{(n)}$

$$Z_t = \sum_{n=1}^{N} \omega_{t-1}^{(n)} e^{-\alpha_t y^{(n)} h_t\left(\boldsymbol{x}^{(n)}\right)}$$

$$= \sum_{y^{(n)} = h_t\left(\boldsymbol{x}^{(n)}\right)} \omega_{t-1}^{(n)} e^{-\alpha_t} + \sum_{y^{(n)} \neq h_t\left(\boldsymbol{x}^{(n)}\right)} \omega_{t-1}^{(n)} e^{\alpha_t}$$

$$= e^{-\alpha_t} \sum_{y^{(n)} = h_t\left(\boldsymbol{x}^{(n)}\right)} \omega_{t-1}^{(n)} + e^{\alpha_t} \sum_{y^{(n)} \neq h_t\left(\boldsymbol{x}^{(n)}\right)} \omega_{t-1}^{(n)}$$

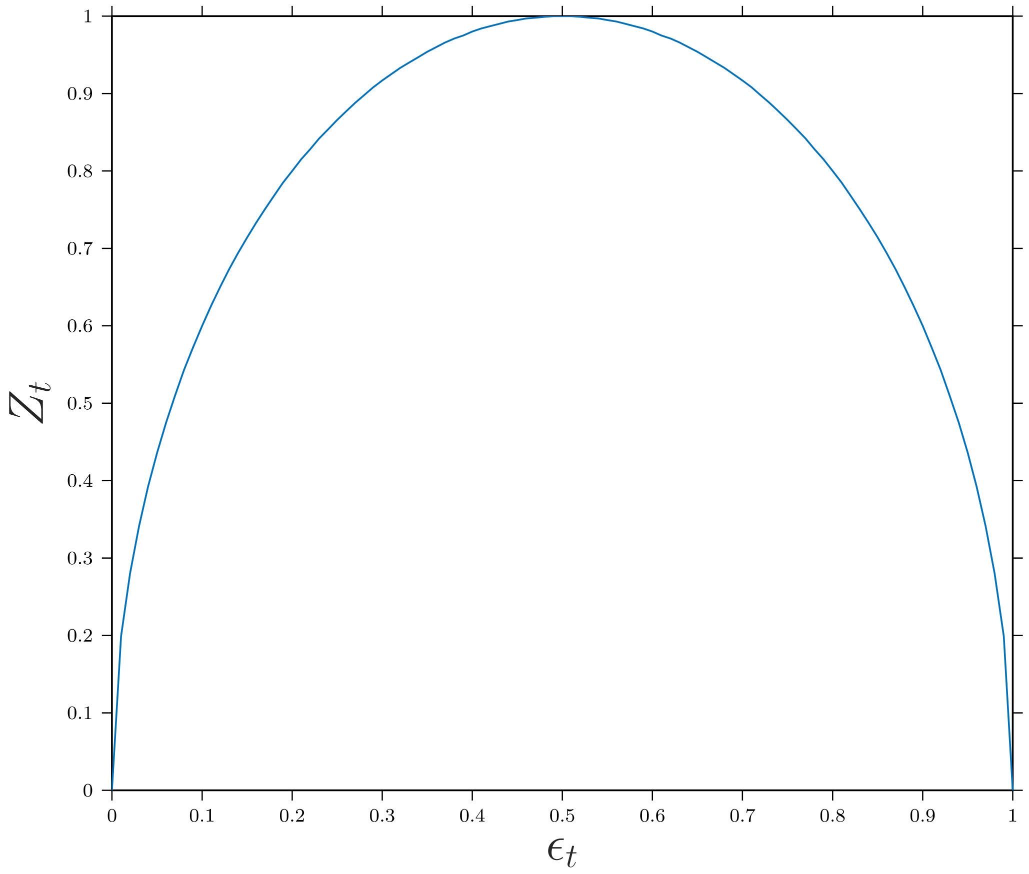
$$= e^{-\alpha_t}(1 - \epsilon_t) + e^{\alpha_t}\epsilon_t$$

$$= e^{-\frac{1}{2}\log\left(\frac{1-\epsilon_t}{\epsilon_t}\right)}(1 - \epsilon_t) + e^{\frac{1}{2}\log\left(\frac{1-\epsilon_t}{\epsilon_t}\right)}\epsilon_t$$

$$= \sqrt{\epsilon_t(1 - \epsilon_t)} + \sqrt{\epsilon_t(1 - \epsilon_t)} = 2\sqrt{\epsilon_t(1 - \epsilon_t)}$$

Z_t

$$Z_t = \sum_{n=1}^{N} \omega_{t-1}^{(n)} e^{-\alpha_t y^{(n)} h_t(\boldsymbol{x}^{(n)})} = 2\sqrt{\epsilon_t(1-\epsilon_t)} < 1 \text{ if } \epsilon_t < \frac{1}{2}$$



Training Error

$$\frac{1}{N}\sum_{n=1}^{N}\mathbb{1}\left(y^{(n)} \neq g_T\left(\boldsymbol{x}^{(n)}\right)\right) \leq \frac{1}{N}\sum_{n=1}^{N} e^{\left(-y^{(n)} H_T\left(\boldsymbol{x}^{(n)}\right)\right)}$$

$$= \prod_{t=1}^{T} Z_t$$

$$= \prod_{t=1}^{T} 2\sqrt{\epsilon_t(1-\epsilon_t)} \rightarrow 0 \text{ as T} \rightarrow \infty$$

$$\left(\text{as long as } \epsilon_t < \frac{1}{2}\ \forall\, t\right)$$

True Error (Freund & Schapire, 1995)

- For AdaBoost, with high probability:

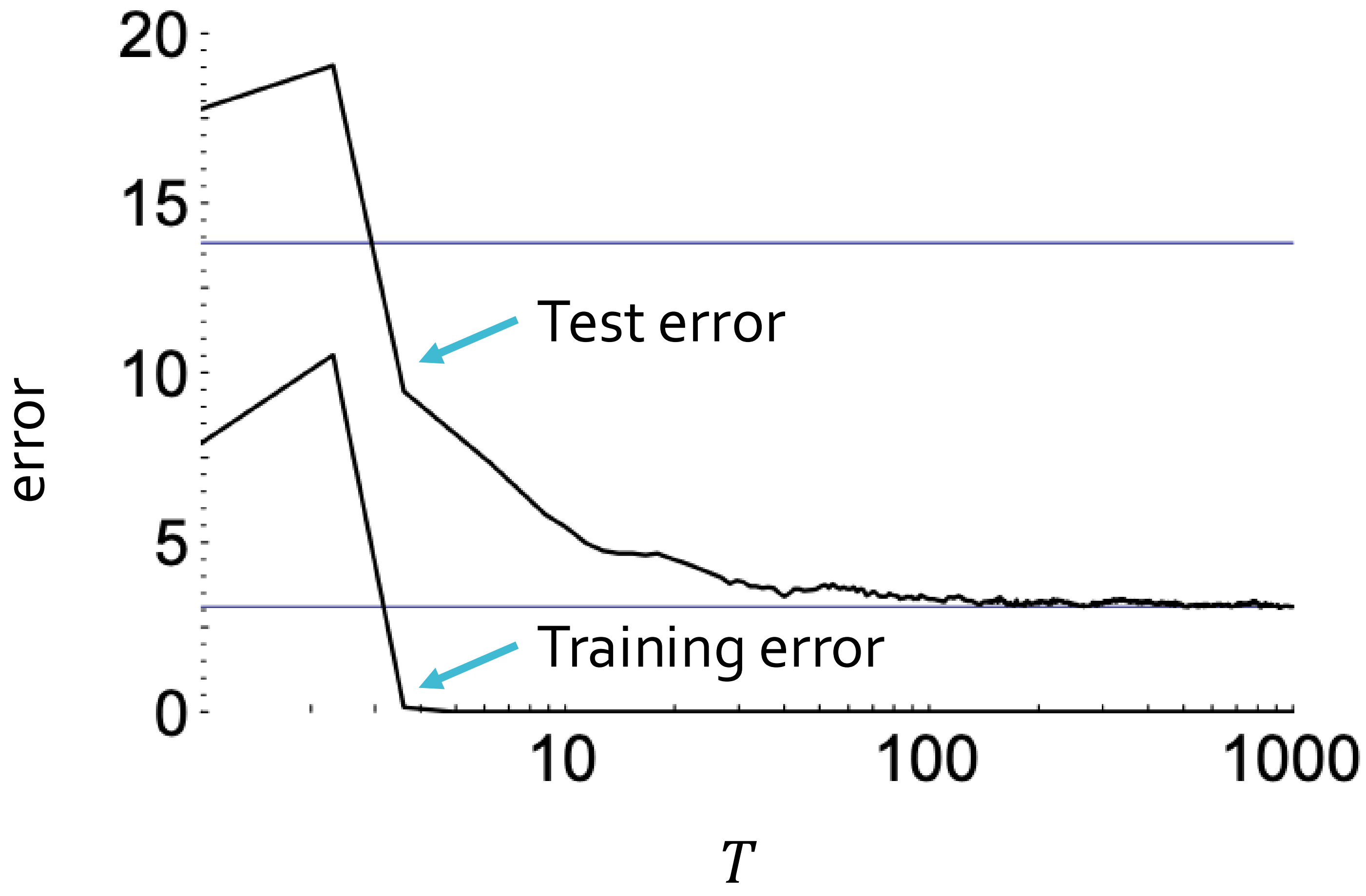
$$\text{True Error} \leq \text{Training Error} + \tilde{O}\left(\sqrt{\frac{d_{vc}(\mathcal{H})T}{N}}\right)$$

where $d_{vc}(\mathcal{H})$ is the VC-dimension of the weak learners and T is the number of weak learners.

- Empirical results indicate that increasing T does not lead to overfitting as this bound would suggest!

Source: http://rob.schapire.net/papers/FreundSc95.pdf

Test Error (Schapire, 1989)



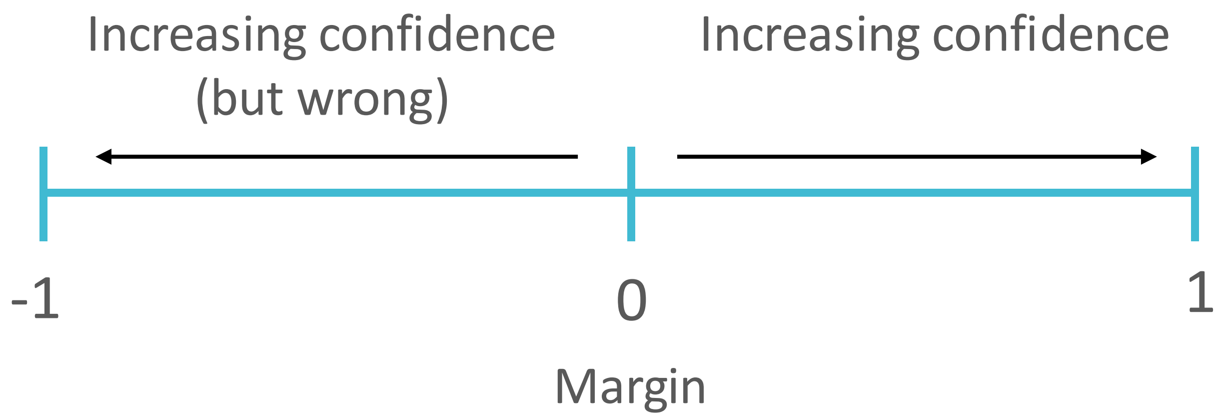
Source: http://rob.schapire.net/papers/msri.pdf

Margins

- The *margin* of training point $\left(\boldsymbol{x}^{(i)}, y^{(i)}\right)$ is defined as:

$$m\left(\boldsymbol{x}^{(i)}, y^{(i)}\right) = \frac{y^{(i)} \sum_{t=1}^{T} \alpha_t h_t\left(\boldsymbol{x}^{(i)}\right)}{\sum_{t=1}^{T} \alpha_t}$$

- The margin can be interpreted as how confident g_T is in its prediction: the bigger the margin, the more confident.



True Error (Schapire, Freund et al., 1998)

- For AdaBoost, with high probability:

$$\text{True Error} \leq \frac{1}{N}\sum_{i=1}^{N} [\![m(\boldsymbol{x}^{(i)}, y^{(i)}) \leq \epsilon]\!] + \tilde{O}\left(\sqrt{\frac{d_{vc}(\mathcal{H})}{N\epsilon^2}}\right)$$

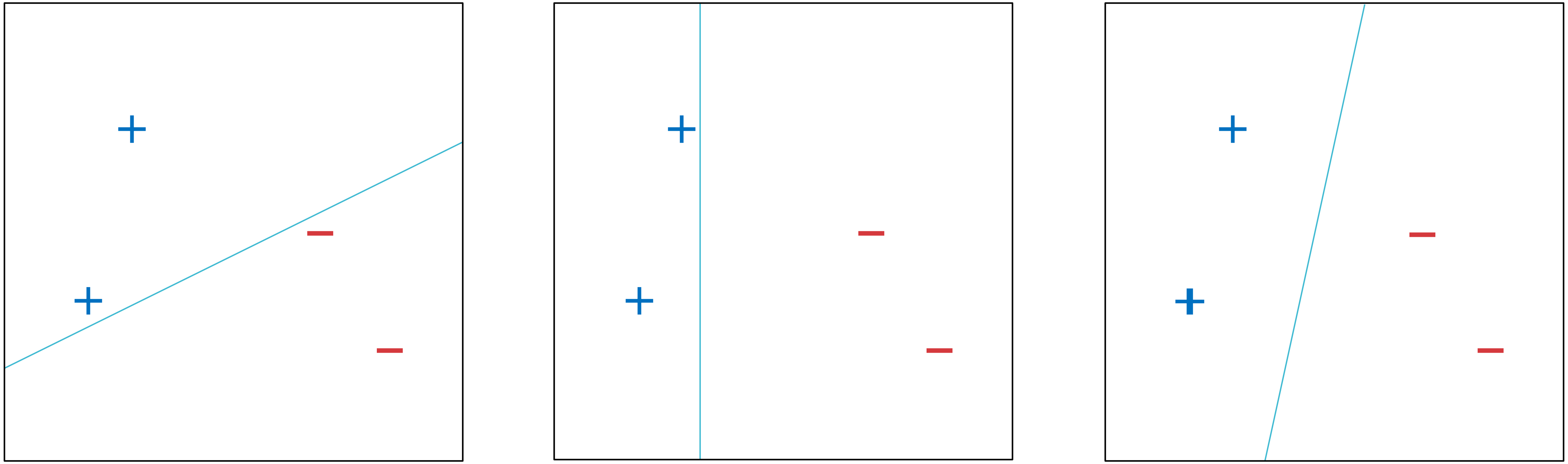
where $d_{vc}(\mathcal{H})$ is the VC-dimension of the weak learners and $\epsilon > 0$ is a tolerance parameter.

- Even after AdaBoost has driven the training error to 0, it continues to target the "training margin"

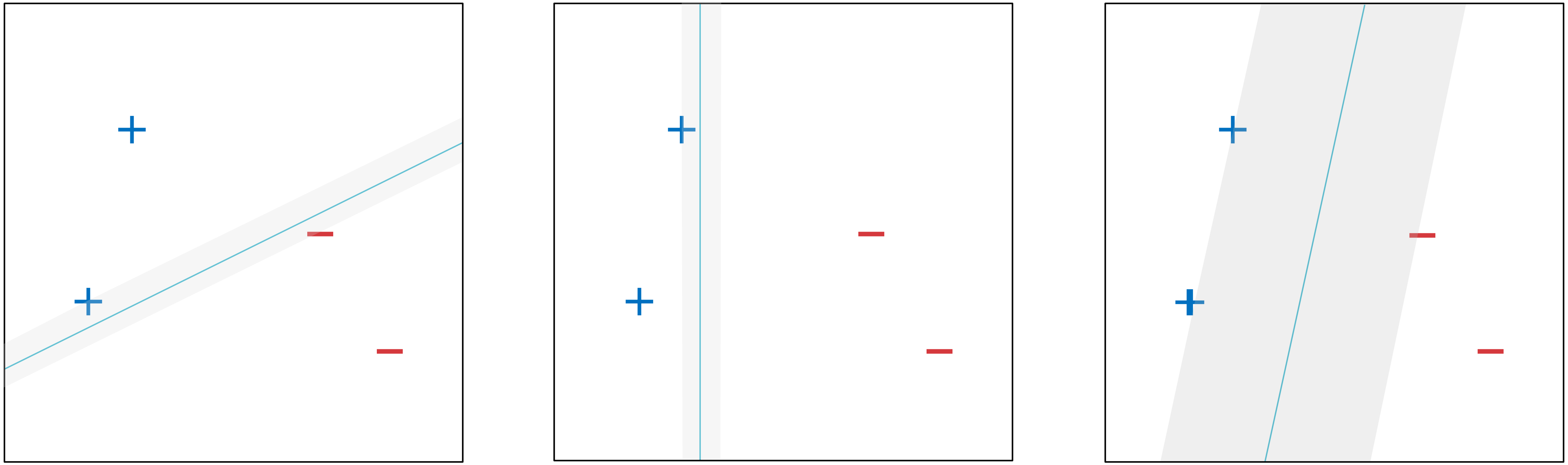
Source: http://rob.schapire.net/papers/SchapireFrBaLe98.pdf

Key Takeaways

- Boosting targets simple models, i.e., weak learners
- Greedily minimizes the exponential loss, an upper bound of the classification error
- Theoretical (and empirical) results show resilience to overfitting by targeting training margin



In-class Poll:
Which linear separator is best?



Which linear separator is best?

Maximal Margin Linear Separators

- The margin of a linear separator is the distance between it and the nearest training data point
- Questions:
  1. How can we efficiently find a maximal-margin linear separator?
  2. Why are linear separators with larger margins better?
  3. What can we do if the data is not linearly separable?

Recall: Hyperplanes

- For linear models, decision boundaries are D-dimensional ***hyperplanes*** defined by a weight vector, $[b, \boldsymbol{w}]$

$$\boldsymbol{w}^T \boldsymbol{x} + b = 0$$

- Problem: there are infinitely many weight vectors that describe the same hyperplane
    - $x_1 + 2x_2 + 2 = 0$ is the same line as $2x_1 + 4x_2 + 4 = 0$, which is the same line as $1000000x_1 + 2000000x_2 + 2000000 = 0$
- Solution: normalize weight vectors *w.r.t. the training data*

Normalizing Hyperplanes

- Given a dataset $\mathcal{D} = \left\{\left(\boldsymbol{x}^{(i)}, y^{(i)}\right)\right\}_{i=1}^{N}$ where $y \in \{-1, +1\}$, $\hat{y} = \text{sign}(\boldsymbol{w}^T \boldsymbol{x} + b)$ is a valid **linear separator** if

$$y^{(i)}\left(\boldsymbol{w}^T \boldsymbol{x}^{(i)} + b\right) > 0 \; \forall \left(\boldsymbol{x}^{(i)}, y^{(i)}\right) \in \mathcal{D}$$

- For SVMs, we're *only* going to consider **linear separators** in

$$\mathcal{H} = \left\{\hat{y} = \text{sign}(\boldsymbol{w}^T \boldsymbol{x} + b) : \min_{\left(\boldsymbol{x}^{(i)}, y^{(i)}\right) \in \mathcal{D}} y^{(i)}\left(\boldsymbol{w}^T \boldsymbol{x}^{(i)} + b\right) = 1\right\}$$

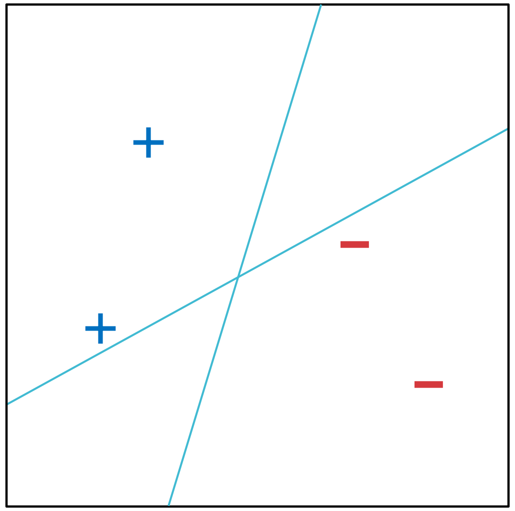
- If $\hat{y} = \text{sign}(\boldsymbol{w}^T \boldsymbol{x} + b)$ is a linear separator, then

$\hat{y} = \text{sign}\left(\frac{\boldsymbol{w}^T}{\rho} \boldsymbol{x} + \frac{b}{\rho}\right) \in \mathcal{H}$ where

$$\rho = \min_{\left(\boldsymbol{x}^{(i)}, y^{(i)}\right) \in \mathcal{D}} y^{(i)}\left(\boldsymbol{w}^T \boldsymbol{x}^{(i)} + b\right)$$

Normalizing Hyperplanes: Example

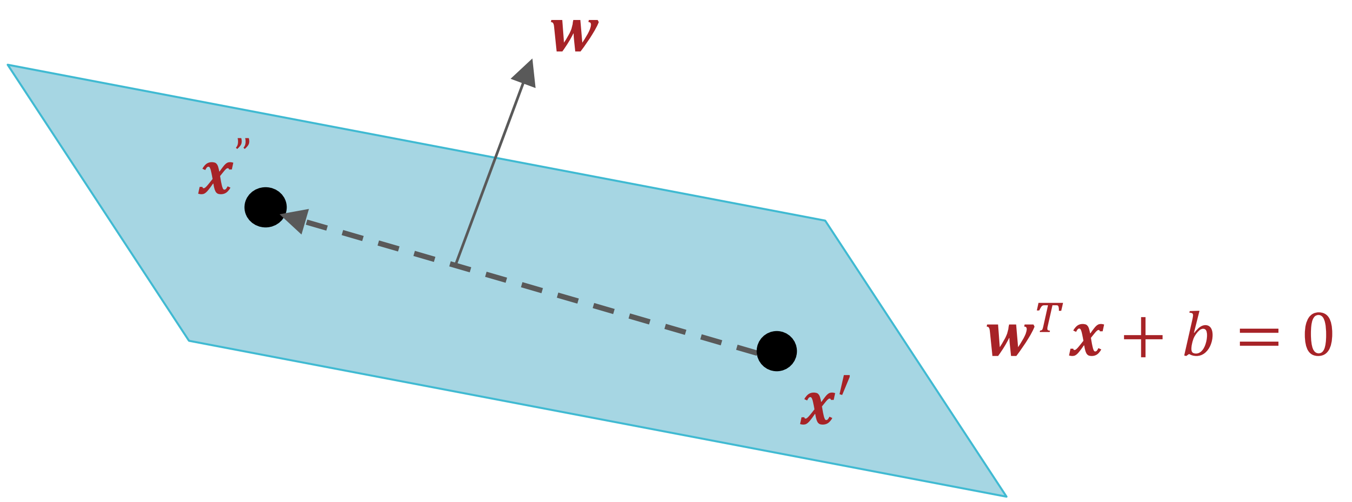
b	w_1	w_2	
-0.2	-0.6	1	$\notin \mathcal{H}$
-0.4	-1.2	2	$\notin \mathcal{H}$
-2	-6	10	$\notin \mathcal{H}$
-10	-30	50	$\in \mathcal{H}$
0.2	-0.6	0.2	$\notin \mathcal{H}$
0.1	-0.3	0.1	$\notin \mathcal{H}$
1	-3	1	$\notin \mathcal{H}$
2	-6	2	$\in \mathcal{H}$



x_1	x_2	y	$y(\boldsymbol{w}^T\boldsymbol{x} + b)$
0.2	0.4	+1	1.6
0.3	0.8	+1	1.8
0.7	0.6	-1	1
0.8	0.3	-1	2.2

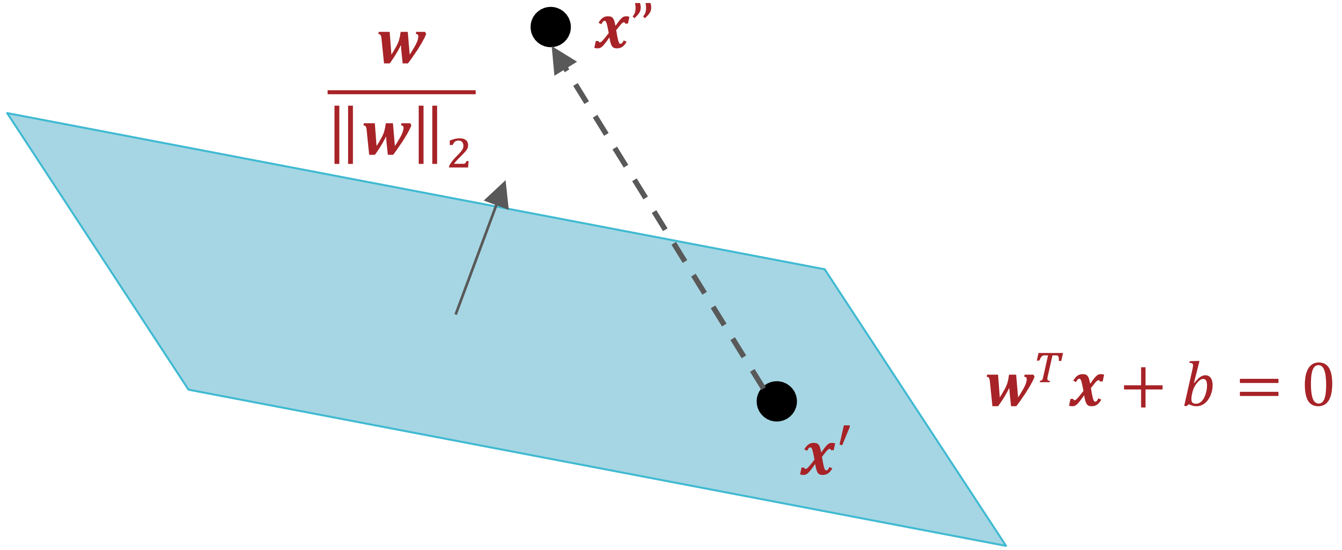
Computing the Margin

- Claim: $\boldsymbol{w}$ is orthogonal to the hyperplane $\boldsymbol{w}^T\boldsymbol{x} + b = 0$ (the decision boundary)
- A vector is orthogonal to a hyperplane if it is orthogonal to every vector in that hyperplane
- Vectors $\boldsymbol{\alpha}$ and $\boldsymbol{\beta}$ are orthogonal if $\boldsymbol{\alpha}^T\boldsymbol{\beta} = 0$



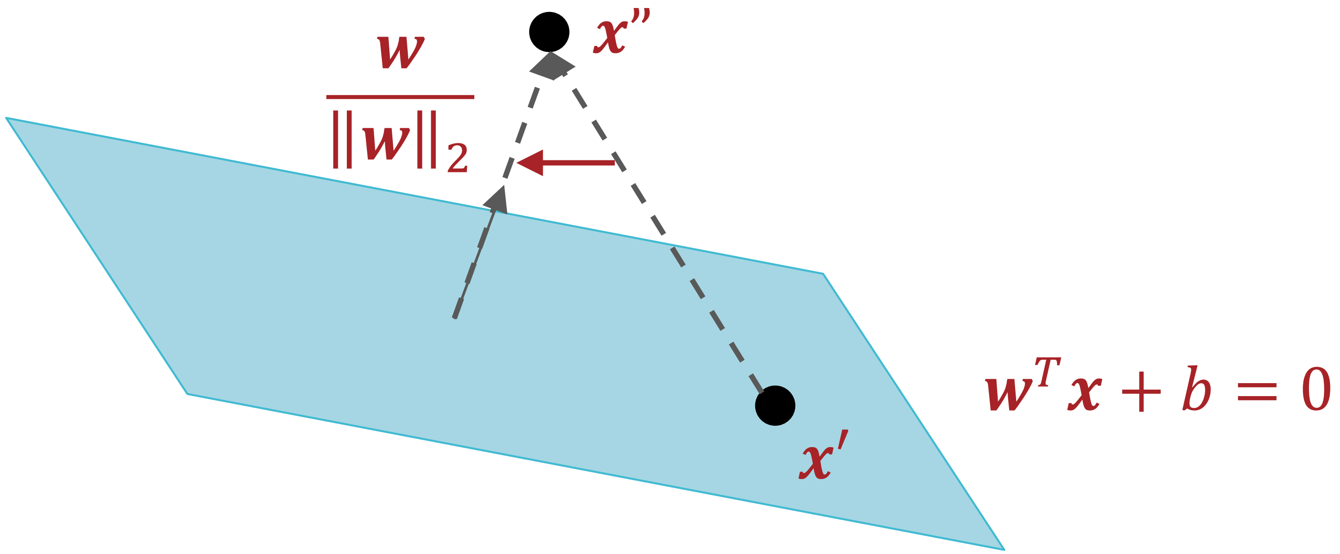
Computing the Margin

- Let $\boldsymbol{x}'$ be an arbitrary point on the hyperplane $\boldsymbol{w}^T\boldsymbol{x} + b = 0$ and let $\boldsymbol{x}''$ be an arbitrary point
- The distance between $\boldsymbol{x}''$ and $\boldsymbol{w}^T\boldsymbol{x} + b = 0$ is equal to the magnitude of the projection of $\boldsymbol{x}'' - \boldsymbol{x}'$ onto $\frac{\boldsymbol{w}}{\|\boldsymbol{w}\|_2}$, the unit vector orthogonal to the hyperplane



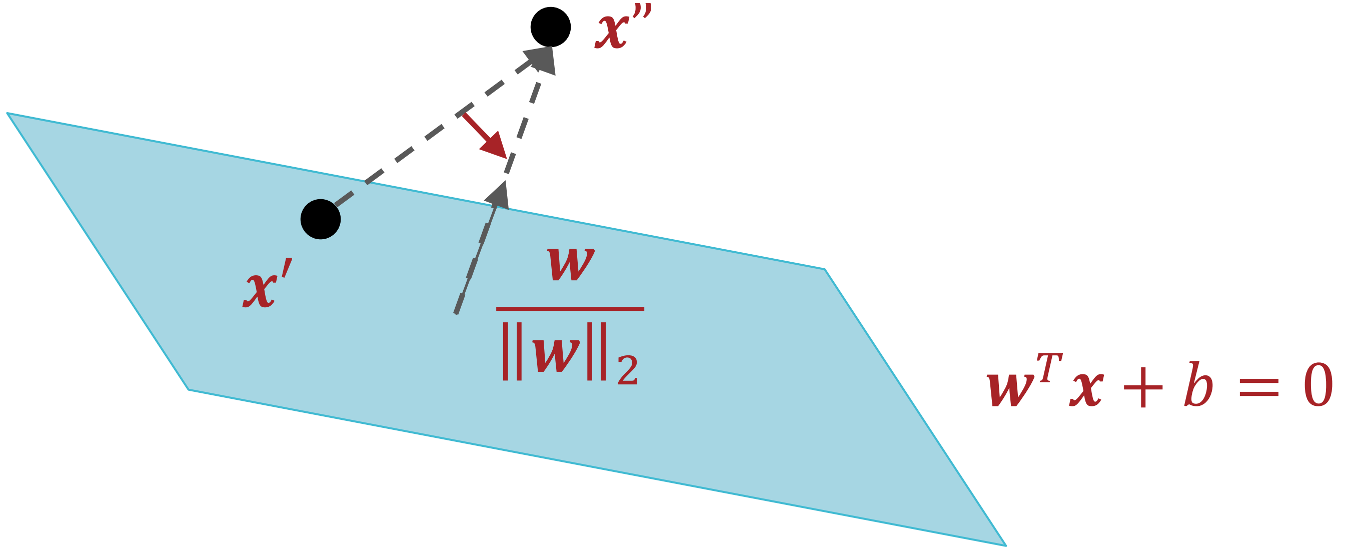
Computing the Margin

- Let $\boldsymbol{x}'$ be an arbitrary point on the hyperplane $\boldsymbol{w}^T\boldsymbol{x} + b = 0$ and let $\boldsymbol{x}''$ be an arbitrary point
- The distance between $\boldsymbol{x}''$ and $\boldsymbol{w}^T\boldsymbol{x} + b = 0$ is equal to the magnitude of the projection of $\boldsymbol{x}'' - \boldsymbol{x}'$ onto $\frac{\boldsymbol{w}}{\|\boldsymbol{w}\|_2}$, the unit vector orthogonal to the hyperplane



Computing the Margin

- Let $\boldsymbol{x}'$ be an arbitrary point on the hyperplane $\boldsymbol{w}^T\boldsymbol{x} + b = 0$ and let $\boldsymbol{x}''$ be an arbitrary point
- The distance between $\boldsymbol{x}''$ and $\boldsymbol{w}^T\boldsymbol{x} + b = 0$ is equal to the magnitude of the projection of $\boldsymbol{x}'' - \boldsymbol{x}'$ onto $\frac{\boldsymbol{w}}{\|\boldsymbol{w}\|_2}$, the unit vector orthogonal to the hyperplane



Computing the Margin

- Let $\boldsymbol{x}'$ be an arbitrary point on the hyperplane $h(\boldsymbol{x}) = \boldsymbol{w}^T\boldsymbol{x} + b = 0$ and let $\boldsymbol{x}''$ be an arbitrary point
- The distance between $\boldsymbol{x}''$ and $h(\boldsymbol{x}) = \boldsymbol{w}^T\boldsymbol{x} + b = 0$ is equal to the magnitude of the projection of $\boldsymbol{x}'' - \boldsymbol{x}'$ onto $\frac{\boldsymbol{w}}{\|\boldsymbol{w}\|_2}$, the unit vector orthogonal to the hyperplane

$$d(\boldsymbol{x}'', h) = \left|\frac{\boldsymbol{w}^T(\boldsymbol{x}'' - \boldsymbol{x}')}{\|\boldsymbol{w}\|_2}\right| = \frac{|\boldsymbol{w}^T\boldsymbol{x}'' - \boldsymbol{w}^T\boldsymbol{x}'|}{\|\boldsymbol{w}\|_2}$$

$$= \frac{|\boldsymbol{w}^T\boldsymbol{x}'' + b|}{\|\boldsymbol{w}\|_2}$$

Computing the Margin

- The margin of a linear separator is the distance between it and the nearest training data point

$$\min_{(\boldsymbol{x}^{(i)},y^{(i)})\in\mathcal{D}} d\left(\boldsymbol{x}^{(i)},h\right) = \min_{(\boldsymbol{x}^{(i)},y^{(i)})\in\mathcal{D}} \frac{\left|\boldsymbol{w}^T\boldsymbol{x}^{(i)}+b\right|}{\|\boldsymbol{w}\|_2}$$

$$= \frac{1}{\|\boldsymbol{w}\|_2}\min_{(\boldsymbol{x}^{(i)},y^{(i)})\in\mathcal{D}} \left|\boldsymbol{w}^T\boldsymbol{x}^{(i)}+b\right|$$

$$= \frac{1}{\|\boldsymbol{w}\|_2}\min_{(\boldsymbol{x}^{(i)},y^{(i)})\in\mathcal{D}} y^{(i)}\left(\boldsymbol{w}^T\boldsymbol{x}^{(i)}+b\right)$$

$$= \frac{1}{\|\boldsymbol{w}\|_2}$$

Maximizing the Margin

$$\text{maximize } \frac{1}{\|\boldsymbol{w}\|_2}$$

$$\text{subject to } \min_{\left(\boldsymbol{x}^{(i)}, y^{(i)}\right) \in \mathcal{D}} y^{(i)}\left(\boldsymbol{w}^T \boldsymbol{x}^{(i)} + b\right) = 1$$

$$\Updownarrow$$

$$\text{minimize } \|\boldsymbol{w}\|_2$$

$$\text{subject to } \min_{\left(\boldsymbol{x}^{(i)}, y^{(i)}\right) \in \mathcal{D}} y^{(i)}\left(\boldsymbol{w}^T \boldsymbol{x}^{(i)} + b\right) = 1$$

$$\Updownarrow$$

$$\text{minimize } \frac{1}{2}\|\boldsymbol{w}\|_2^2$$

$$\text{subject to } \min_{\left(\boldsymbol{x}^{(i)}, y^{(i)}\right) \in \mathcal{D}} y^{(i)}\left(\boldsymbol{w}^T \boldsymbol{x}^{(i)} + b\right) = 1$$

$$\Updownarrow$$

$$\text{minimize } \frac{1}{2}\boldsymbol{w}^T \boldsymbol{w}$$

$$\text{subject to } y^{(i)}\left(\boldsymbol{w}^T \boldsymbol{x}^{(i)} + b\right) \geq 1 \ \forall \left(\boldsymbol{x}^{(i)}, y^{(i)}\right) \in \mathcal{D}$$