Boosting Approach to ML

Perceptron, Margins, Kernels

Maria-Florina Balcan

03/18/2015

Recap from last time: Boosting

- General method for improving the accuracy of any given learning algorithm.
- Works by creating a series of challenge datasets s.t. even modest performance on these can be used to produce an overall high-accuracy predictor.
- Adaboost one of the top 10 ML algorithms.
  - Works amazingly well in practice.
  - Backed up by solid foundations.

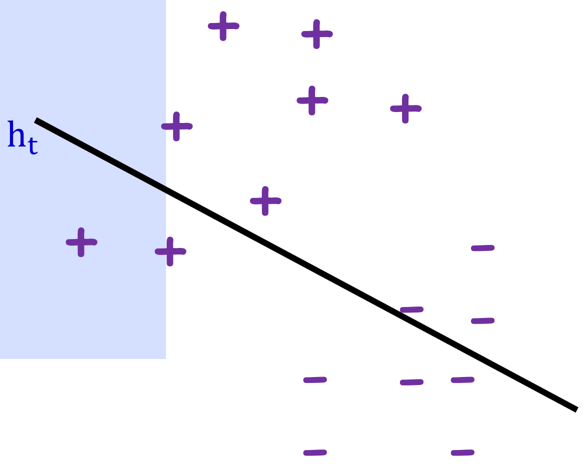
Adaboost (Adaptive Boosting)

Input: S={(x_1, y_1), …,(x_m, y_m)}; $x_i \in X, y_i \in Y = \{-1,1\}$

weak learning algo *A* (e.g., Naïve Bayes, decision stumps)

- For t=1,2, … ,T
  - Construct D_t on $\{x_1, \dots, x_m\}$
  - Run *A* on D_t producing $h_t: X \rightarrow \{-1,1\}$

Output $H_{\text{final}}(x) = \text{sign}(\sum_{t=1} \alpha_t h_t(x))$



- D_1 uniform on $\{x_1, \dots, x_m\}$ [i.e., $D_1(i) = \frac{1}{m}$]
- Given D_t and h_t set

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

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

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

$$\alpha_t = \frac{1}{2} \ln\left(\frac{1-\epsilon_t}{\epsilon_t}\right) > 0$$

D_{t+1} puts half of weight on examples x_i where h_t is incorrect & half on examples where h_t is correct

Nice Features of Adaboost

- Very general: a meta-procedure, it can use any weak learning algorithm!!! (e.g., Naïve Bayes, decision stumps)
- Very fast (single pass through data each round) & simple to code, no parameters to tune.
- Grounded in rich theory.

Analyzing Training Error

Theorem $\epsilon_t = 1/2 - \gamma_t$ (error of h_t over D_t)

$$err_S(H_{final}) \leq \exp\left[-2\sum_t \gamma_t^2\right]$$

So, if $\forall t, \gamma_t \geq \gamma > 0$, then $err_S(H_{final}) \leq \exp[-2\,\gamma^2 T]$

The training error drops exponentially in T!!!

To get $err_S(H_{final}) \leq \epsilon$, need only $T = O\left(\frac{1}{\gamma^2}\log\left(\frac{1}{\epsilon}\right)\right)$ rounds

Adaboost is adaptive

- Does not need to know γ or T a priori
- Can exploit $\gamma_t \gg \gamma$

Generalization Guarantees

Theorem $err_S(H_{final}) \leq \exp\left[-2\sum_t \gamma_t^2\right]$ where $\epsilon_t = 1/2 - \gamma_t$

How about generalization guarantees?

Original analysis [Freund&Schapire'97]

- H space of weak hypotheses; d=VCdim(H)

  H_{final} is a weighted vote, so the hypothesis class is:

  G={all fns of the form $\mathrm{sign}(\sum_{t=1}^{T} \alpha_t h_t(x))$ }

$p = VCdim(G)?$

Theorem [Freund&Schapire'97]

$\forall\, g \in G, err(g) \leq err_S(g) + \tilde{O}\left(\sqrt{\frac{Td}{m}}\right)$ T= # of rounds

Key reason: $\mathrm{VC}dim(G) = \tilde{O}(dT)$ plus typical VC bounds.

H: infinite.

$c^* \notin H$

$err_D(h) \leq err_S(h) + O\left(\sqrt{\frac{d}{m}\left(\ln\frac{m}{d} + \ln\frac{1}{\delta}\right)}\right)$

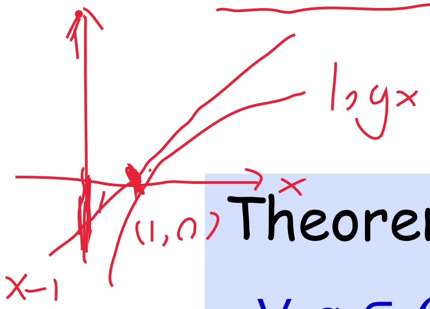
$1-\frac{1}{x} \le \log x \le x-1$

$\alpha > 0$,

$\log(\alpha x) \le \alpha x - 1 \Rightarrow \log x \le \alpha x - \log\alpha - 1$

Generalization Guarantees

$H[p] = O(m^d)$

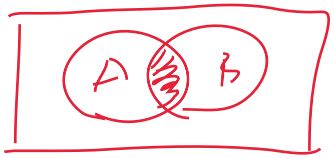


Theorem [Freund&Schapire'97]

$\forall g \in G, err(g) \le err_S(g) + \underbrace{\tilde{O}\left(\sqrt{\frac{Td}{m}}\right)}$ T= # of rounds

$d := \text{VC dim}(~~H~~)$

$\underline{H[p]}$



$\underline{p = O(dT \ln dT)}$

$p = \text{VC dim}(G)$

$\Sigma^p = \underline{G[p]} \le (H[p])^T = (O(p^d))^T = O(p^{dT})$

shattering

coefficient.

$p \le O(\log_2 p^{dT})$

$= O(dT \log_2 p)$.

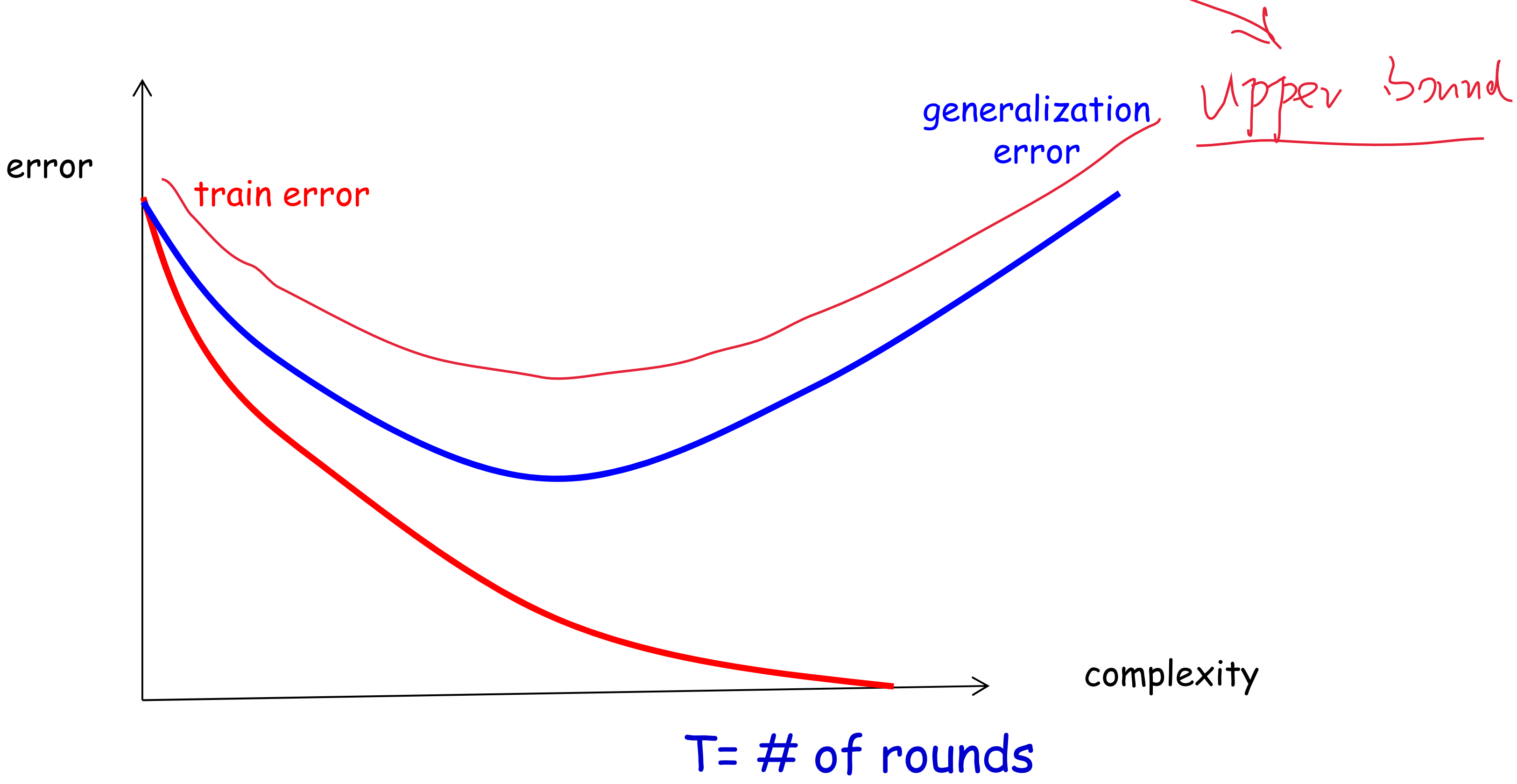
$\le O\, dT\left(\frac{1}{pT}p + \log_2 dT - 1\right) = O(p + dT\log_2 dT - dT)$

$p \le \underline{O(dT\log_2 dT)} = \tilde{O}(dT)$

Generalization Guarantees

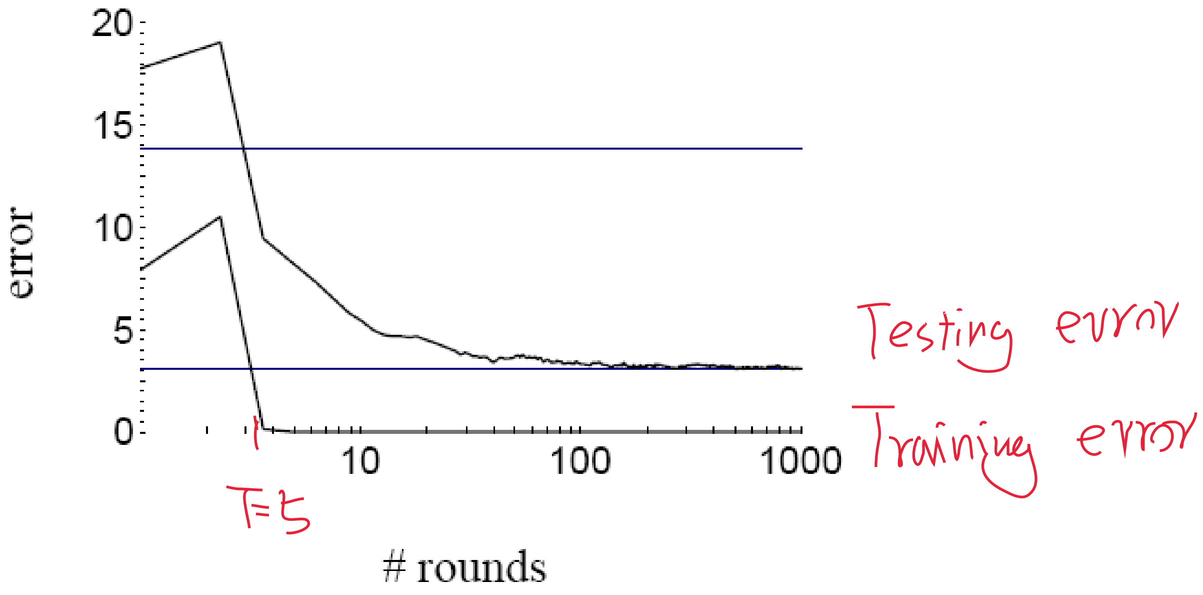
Theorem [Freund&Schapire'97]

$$\forall\, g \in G, err(g) \le err_S(g) + \tilde{O}\left(\sqrt{\frac{Td}{m}}\right)$$ where d=VCdim(H)



Generalization Guarantees

- Experiments showed that the test error of the generated classifier usually does not increase as its size becomes very large.
- Experiments showed that continuing to add new weak learners after correct classification of the training set had been achieved could further improve test set performance!!!



Generalization Guarantees

- Experiments showed that the test error of the generated classifier usually does not increase as its size becomes very large.

- Experiments showed that continuing to add new weak learners after correct classification of the training set had been achieved could further improve test set performance!!!

- These results seem to contradict FS'97 bound and Occam's razor (in order achieve good test error the classifier should be as simple as possible)!

$$\forall\, g \in G, err(g) \le err_S(g) + \tilde{O}\left(\sqrt{\frac{Td}{m}}\right)$$

How can we explain the experiments?

R. Schapire, Y. Freund, P. Bartlett, W. S. Lee. present in "*Boosting the margin: A new explanation for the effectiveness of voting methods*" a nice theoretical explanation.

Key Idea:

Training error does not tell the whole story.

We need also to consider the classification confidence!!

$$H_{final}(x) = sign\left(\underline{\sum_{t=1}^{T} \alpha_t h_t(x)}\right)$$

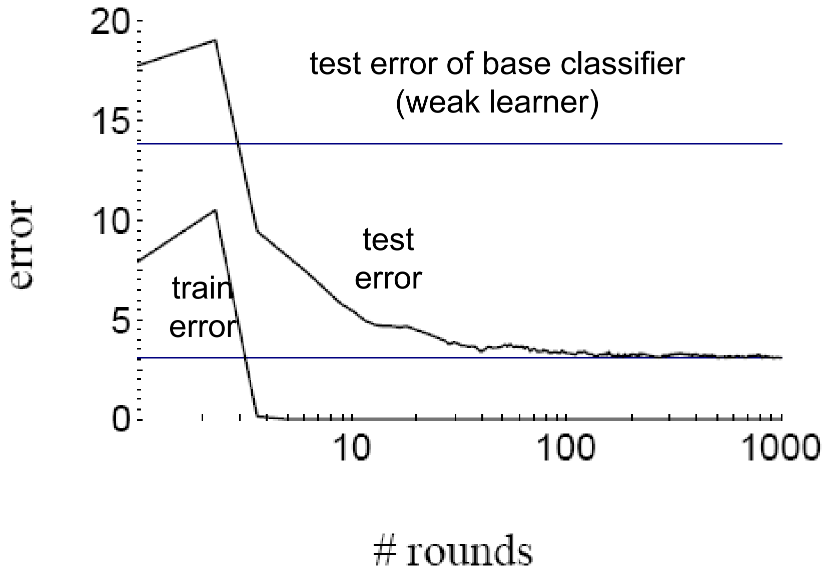
$$f(x) = \overset{+}{0.001}, \quad \overset{+}{0.1}, \quad \overset{+}{1}.$$

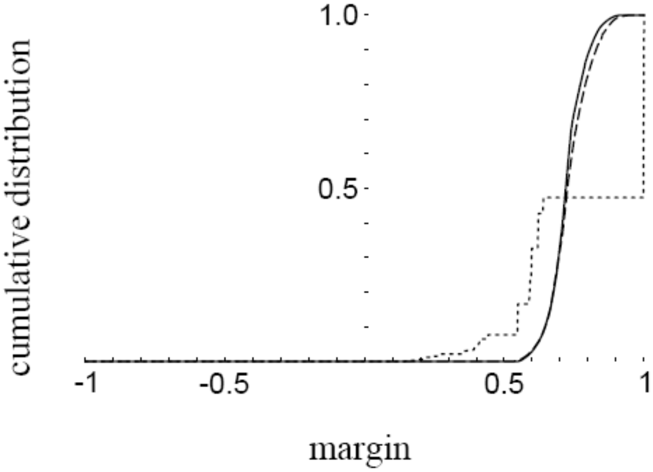
$$f(x) = \sum_{t=1}^{T} \alpha_t h_t(x), \quad \left(\alpha_t \leftarrow \frac{\alpha_t}{\sum_t \alpha_t}\right)$$

Boosting didn't seem to overfit…(!)

…because it turned out to be increasing the *margin* of the classifier

$$y_i f(x_i) \in [-1, 1] \quad (i = 1, 2, \ldots, m)$$





Error Curve, Margin Distr. Graph - Plots from [SFBL98]

Classification Margin

- H space of weak hypotheses. The convex hull of H:

$$co(H) = \{f = \sum_{t=1}^{T} \alpha_t h_t\, , \alpha_t \geq 0, \sum_{t=1}^{T} \alpha_t = 1, h_t \in H\}$$

- Let $f \in co(H), f = \sum_{t=1}^{T} \alpha_t h_t, \alpha_t \geq 0,\ \sum_{t=1}^{T} \alpha_t = 1.$

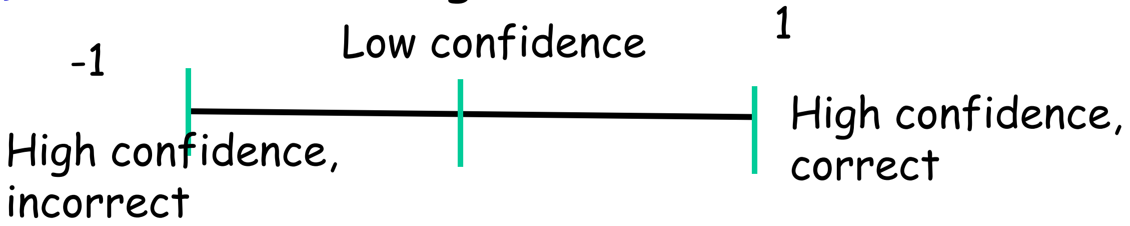
  The majority vote rule H_f given by f (given by $H_f = sign(f(x))$) predicts wrongly on example (x, y) iff $yf(x) \leq 0.$

Definition: margin of H_f (or of f) on example (x, y) to be $yf(x)$.

$$yf(x) = y \sum_{t=1}^{T} [\alpha_t h_t(x)] = \sum_{t=1}^{T} [y \alpha_t h_t(x)] = \sum_{t: y = h_t(x)} \alpha_t - \sum_{t: y \neq h_t(x)} \alpha_t \in [-1, 1]$$

The margin is positive iff $y = H_f(x)$.

See $|yf(x)| = |f(x)|$ as the strength or the confidence of the vote.



$err_S(g) = Pr_S(y \neq g(x)) = Pr_S(yf(x) \leq 0) < Pr_S(yf(x) \leq \theta)$

$(g(x) = sign(f(x)))$

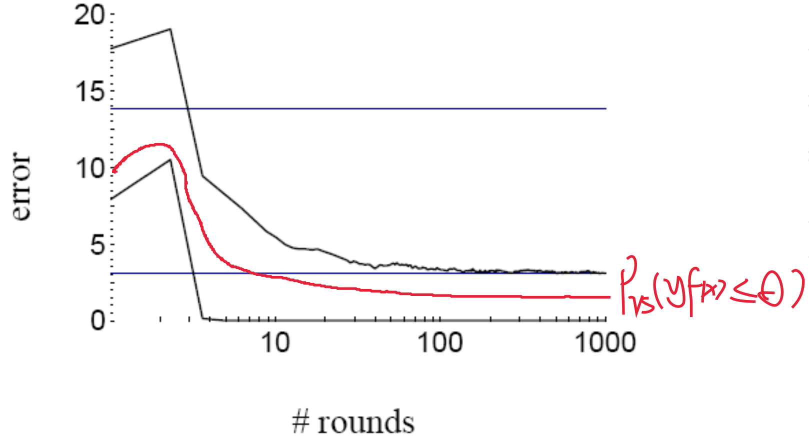
Boosting and Margins

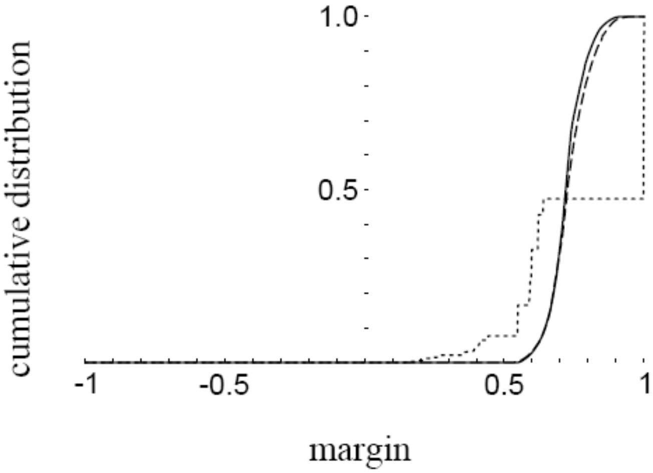
Theorem: VCdim$(H) = d$, then with prob. $\geq 1 - \delta$, $\forall f \in co(H)$, $\forall \theta > 0$,

$\forall g \in \underline{G}$, $err_D(g) = Pr_D(g(x) \neq y)$

$$\underset{D}{\Pr}[\underbrace{yf(x) \leq 0}_{y \neq g(x)}] \leq \boxed{\underset{S}{\Pr}[yf(x) \leq \theta]} + \underline{O\left(\frac{1}{\sqrt{m}}\sqrt{\frac{d \ln^2 \frac{m}{d}}{\theta^2} + \ln \frac{1}{\delta}}\right)} = \tilde{O}\left(\sqrt{\frac{d}{m\theta^2}}\right)$$

Note: bound does **not** depend on T (the # of rounds of boosting), depends only on the complex. of the weak hyp space and the margin!





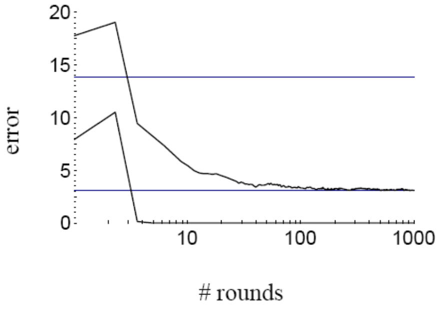
Quiz: according to this slide, explain why adaboost keeps decreasing testing error, even if training error equals to zero.

Boosting and Margins

Theorem: VCdim$(H) = d$, then with prob. $\geq 1 - \delta$, $\forall f \in co(H)$, $\forall \theta > 0$,

$$\Pr_D[yf(x) \leq 0] \leq \Pr_S[yf(x) \leq \theta] + O\left(\frac{1}{\sqrt{m}} \sqrt{\frac{\mathrm{d}\ln^2\frac{m}{d}}{\theta^2} + \ln\frac{1}{\delta}}\right)$$

- If all training examples have large margins, then we can approximate the final classifier by a much smaller classifier.
- Can use this to prove that better margin ➔ smaller test error, regardless of the number of weak classifiers.
- Can also prove that boosting tends to increase the margin of training examples by concentrating on those of smallest margin.
- Although final classifier is getting larger, margins are likely to be increasing, so the final classifier is actually getting closer to a simpler classifier, driving down test error.

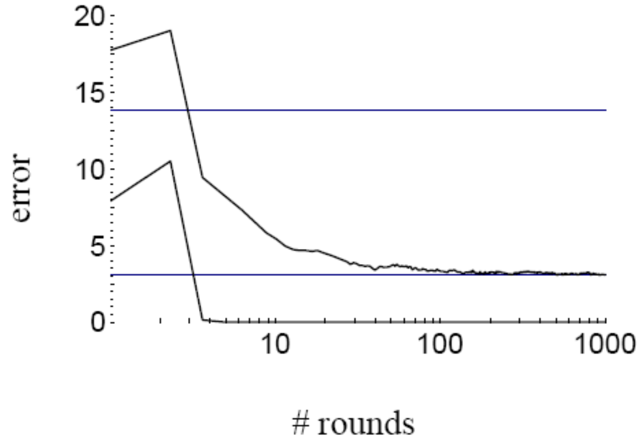


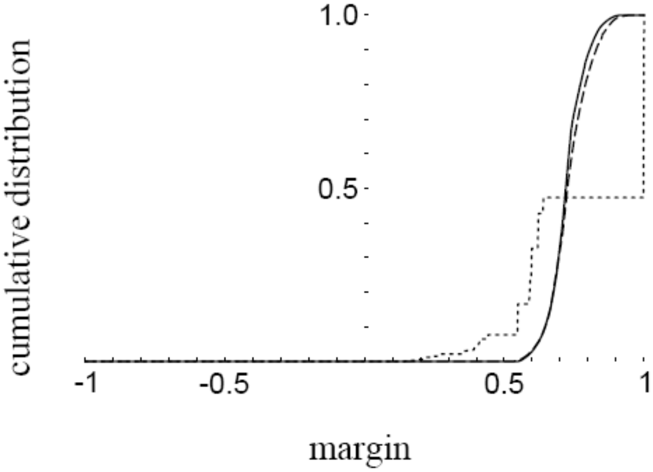
Boosting and Margins

Theorem: VCdim$(H) = d$, then with prob. $\geq 1 - \delta$, $\forall f \in co(H)$, $\forall \theta > 0$,

$$\Pr_D[yf(x) \leq 0] \leq \Pr_S[yf(x) \leq \theta] + O\left(\frac{1}{\sqrt{m}}\sqrt{\frac{\mathrm{d}\ln^2\frac{m}{d}}{\theta^2} + \ln\frac{1}{\delta}}\right)$$

Note: bound does **not** depend on T (the # of rounds of boosting), depends only on the complex. of the weak hyp space and the margin!





Boosting, Adaboost Summary

- Shift in mindset: *goal is now just to find classifiers a bit better than random guessing.*
- Backed up by solid foundations.
- Adaboost work and its variations well in practice with many kinds of data (one of the top 10 ML algos).
- More about classic applications in Recitation.
- Relevant for big data age: quickly focuses on "core difficulties", so well-suited to distributed settings, where data must be communicated efficiently [Balcan-Blum-Fine-Mansour COLT'12].

Interestingly, the usefulness of margin recognized in Machine Learning since late 50's.

Perceptron [Rosenblatt'57] analyzed via geometric (aka L_2, L_2) margin.

Original guarantee in the online learning scenario.

The Perceptron Algorithm

- Online Learning Model
- Margin Analysis
- Kernels

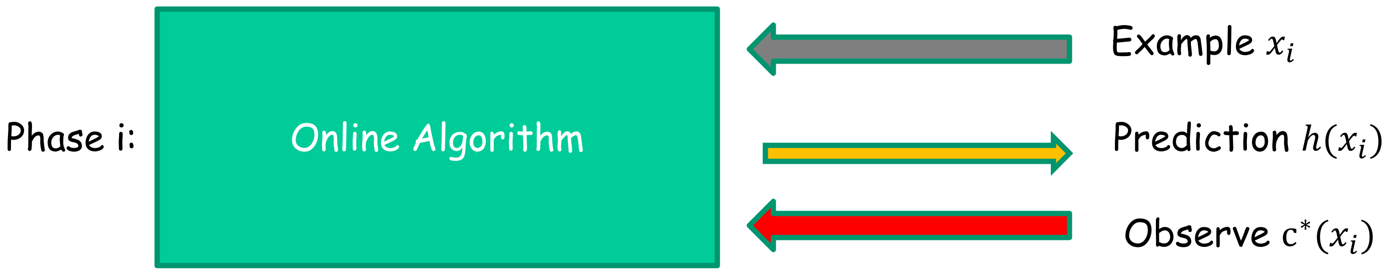
The Online Learning Model

Batch Learning

- Example arrive sequentially.
- We need to make a prediction.

  Afterwards observe the outcome.

For i=1, 2, …, :



Phase i:

Online Algorithm

Example x_i

Prediction $h(x_i)$

Observe $\mathrm{c}^*(x_i)$

Mistake bound model

- Analysis wise, make no distributional assumptions.
- **Goal**: Minimize the number of mistakes.

The Online Learning Model. Motivation

- Email classification (distribution of both spam and regular mail changes over time, but the target function stays fixed - last year's spam still looks like spam).

- Recommendation systems. Recommending movies, etc.

- Predicting whether a user will be interested in a new news article or not.

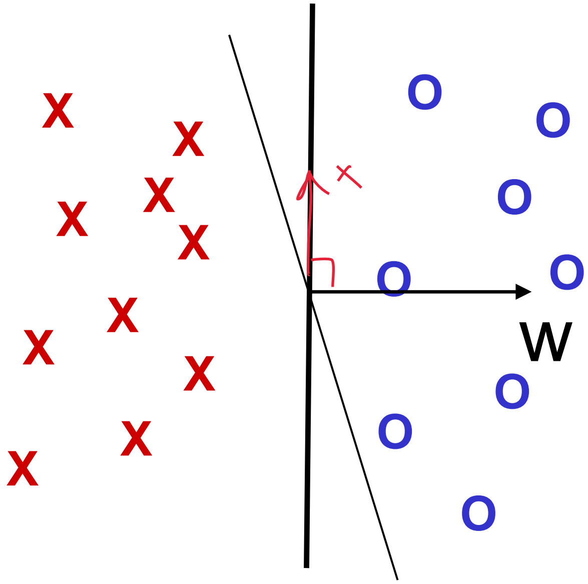
- Add placement in a new market.

$h(x) = \text{sign}(w^T x)$

Binary classification

Linear Separators

- Instance space $\mathrm{X} = \mathrm{R}^{\mathrm{d}}$
- Hypothesis class of linear decision surfaces in R^{d}.
- $\mathrm{h(x)} = \mathrm{w} \cdot \mathrm{x} + \mathrm{w}_0$, if $h(x) \geq 0$, then label x as +, otherwise label it as -



without loss of generality

Claim: WLOG $\mathrm{w}_0 = 0$.

Proof: Can simulate a non-zero threshold with a dummy input feature x_0 that is always set up to 1.

- $x = (x_1, \dots, x_d) \to \tilde{x} = (x_1, \dots, x_d, 1)$
- $\mathrm{w} \cdot \mathrm{x} + \mathrm{w}_0 \geq 0$ iff $(w_1, \dots, w_d, \mathrm{w}_0) \cdot \tilde{x} \geq 0$

where $\mathrm{w} = (w_1, \dots, w_d)$

Linear Separators: Perceptron Algorithm

- Set t=1, start with the all zero vector w_1.
- Given example x, predict positive iff $w_t \cdot x \geq 0$
- On a mistake, update as follows: $w_{t+1} \leftarrow w_t + y x$
  - Mistake on positive, then update $w_{t+1} \leftarrow w_t + x$
  - Mistake on negative, then update $w_{t+1} \leftarrow w_t - x$

Note: w_t is weighted sum of incorrectly classified examples

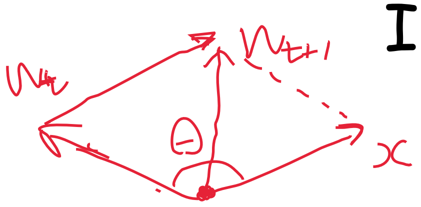
(Sample Sparisity)

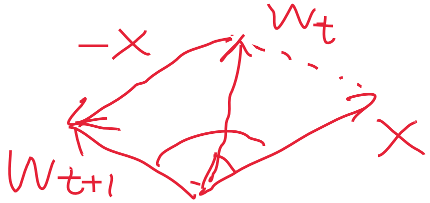
$$w_t = a_{i_1} x_{i_1} + \cdots + a_{i_k} x_{i_k} \quad , \quad a_{i_k} \in \{-1, +1\}$$

$$w_t \cdot x = a_{i_1} \boxed{x_{i_1} \cdot x} + \cdots + a_{i_k} \boxed{x_{i_k} \cdot x}$$

$a \cdot b = \|a\| \cdot \|b\| \cdot \cos\theta$

Important when we talk about kernels.

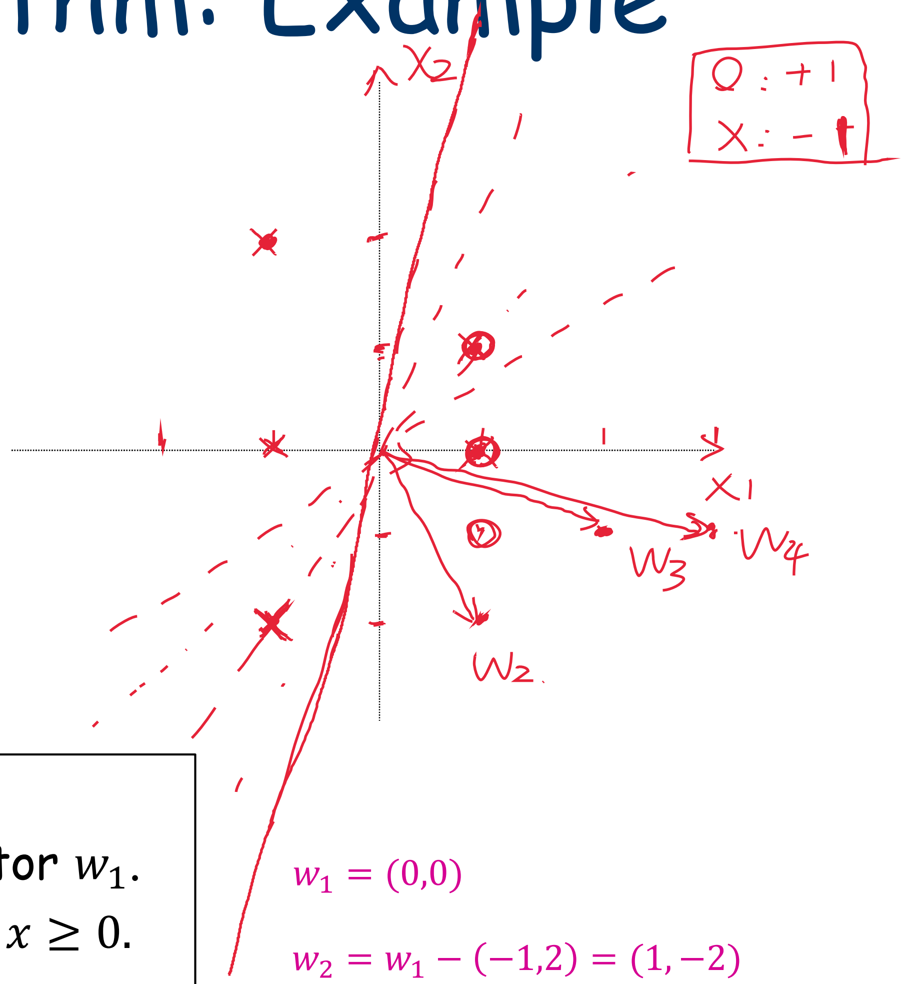




Perceptron Algorithm: Example

Example:

x	y	$\text{sign}(w^T x)$
$(-1,2)$	$-$	+1
$(1,0)$	$+$	+1
$(1,1)$	$+$	-1
$(-1,0)$	$-$	-1
$(-1,-2)$	$-$	+1
$(1,-1)$	$+$	+1



$w_1 = (0,0)$

$w_2 = w_1 - (-1,2) = (1,-2)$

$w_3 = w_2 + (1,1) = (2,-1)$

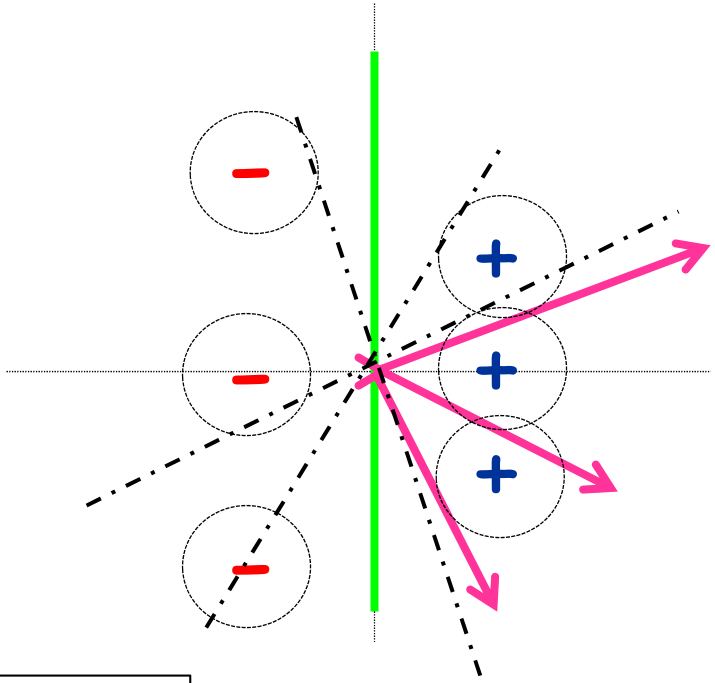
$w_4 = w_3 - (-1,-2) = (3,1)$

Algorithm:

- Set t=1, start with all-zeroes weight vector w_1.
- Given example x, predict positive iff $w_t \cdot x \geq 0$.
  - On a mistake, update as follows:
    - Mistake on positive, update $w_{t+1} \leftarrow w_t + x$
    - Mistake on negative, update $w_{t+1} \leftarrow w_t - x$

Perceptron Algorithm: Example

	x	y	$\text{sign}(w^T x)$	
Example:	$(-1,2)$	$-$	$+$	✗
	$(1,0)$	$+$	$+$	✓
	$(1,1)$	$+$	$-$	✗
	$(-1,0)$	$-$	$-$	✓
	$(-1,-2)$	$-$	$+$	✗
	$(1,-1)$	$+$	$+$	✓



Algorithm:

- Set t=1, start with all-zeroes weight vector w_1.
- Given example x, predict positive iff $w_t \cdot x \geq 0$.
  - On a mistake, update as follows:
    - Mistake on positive, update $w_{t+1} \leftarrow w_t + x$
    - Mistake on negative, update $w_{t+1} \leftarrow w_t - x$

$$w_1 = (0,0)$$

$$w_2 = w_1 - (-1,2) = (1,-2)$$

$$w_3 = w_2 + (1,1) = (2,-1)$$

$$w_4 = w_3 - (-1,-2) = (3,1)$$

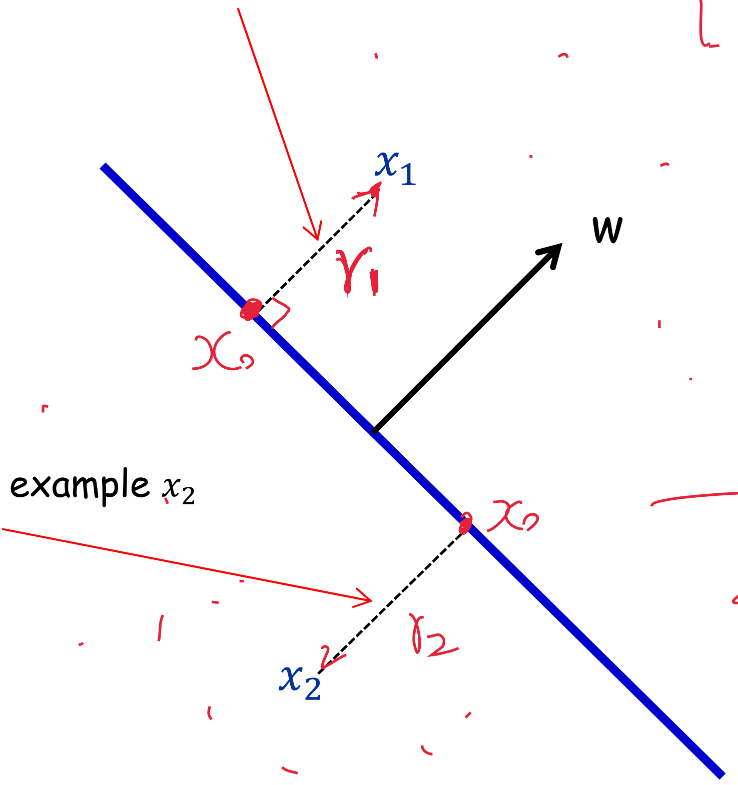
Geometric Margin

Definition: The margin of example x w.r.t. a linear sep. w is the distance from x to the plane $w \cdot x = 0$ (or the negative if on wrong side)

$w^T x = 0$

$$\gamma = \frac{y \cdot x^T w}{||w||}$$

Margin of positive example x_1



Margin of negative example x_2

$$\begin{cases} w^T x_0 = 0 & \textcircled{1} \\ x_1 - x_0 = \gamma_1 \frac{w}{||w||} & \textcircled{2} \end{cases}$$

$$\textcircled{2} \Rightarrow w^T (x_1 - x_0) = \gamma \cdot \frac{w^T w}{||w||}$$

$$\Rightarrow w^T x_1 = \gamma \cdot ||w||$$

$$\Rightarrow \gamma_1 = \frac{w^T x_1}{||w||}$$

$$\begin{cases} w^T x_0 = 0 \\ x_2 - x_0 = -\gamma_2 \frac{w}{||w||} \end{cases}$$

$$\Rightarrow \gamma_2 = -\frac{w^T x_2}{||w||}$$

Geometric Margin

Definition: The margin of example x w.r.t. a linear sep. w is the distance from x to the plane $w \cdot x = 0$ (or the negative if on wrong side)

Geometric Margin

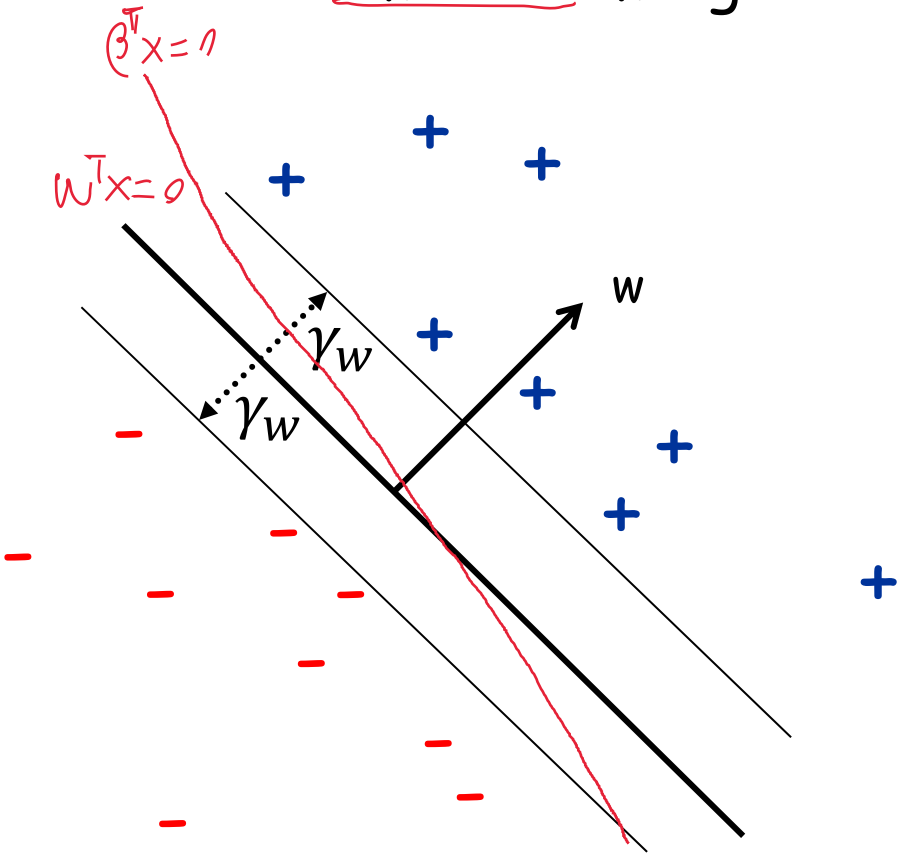
Definition: The margin of example x w.r.t. a linear sep. w is the distance from x to the plane $w \cdot x = 0$ (or the negative if on wrong side)

Definition: The margin γ_w of a set of examples S wrt a linear separator w is the smallest margin over points $x \in S$.

$S = \{(x_i, y_i)\}_{i=1}^{m}$

$$\gamma_w = \min_{i=1,2,\dots,m} \gamma_i = \frac{y_i x_i^T w}{||w||}$$

$(x_i \in S)$



Geometric Margin

Definition: The margin of example x w.r.t. a linear sep. w is the distance from x to the plane $w \cdot x = 0$ (or the negative if on wrong side)

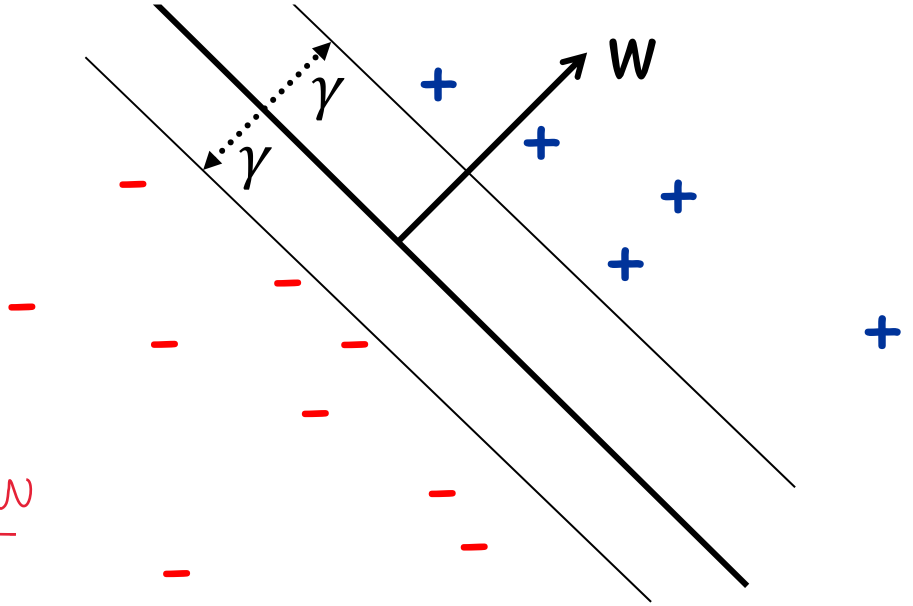
Definition: The margin γ_w of a set of examples S wrt a linear separator w is the smallest margin over points $x \in S$.

Definition: The margin γ of a set of examples S is the maximum γ_w over all linear separators w.

$$\gamma = \max_{w \in \mathbb{R}^d} \gamma_w$$

$$= \max_{w \in \mathbb{R}^d} \min_{x_i \in S} \gamma_i$$

$$= \max_{w \in \mathbb{R}^d} \min_{x_i \in S} \frac{y_i x_i^T w}{\|w\|}$$

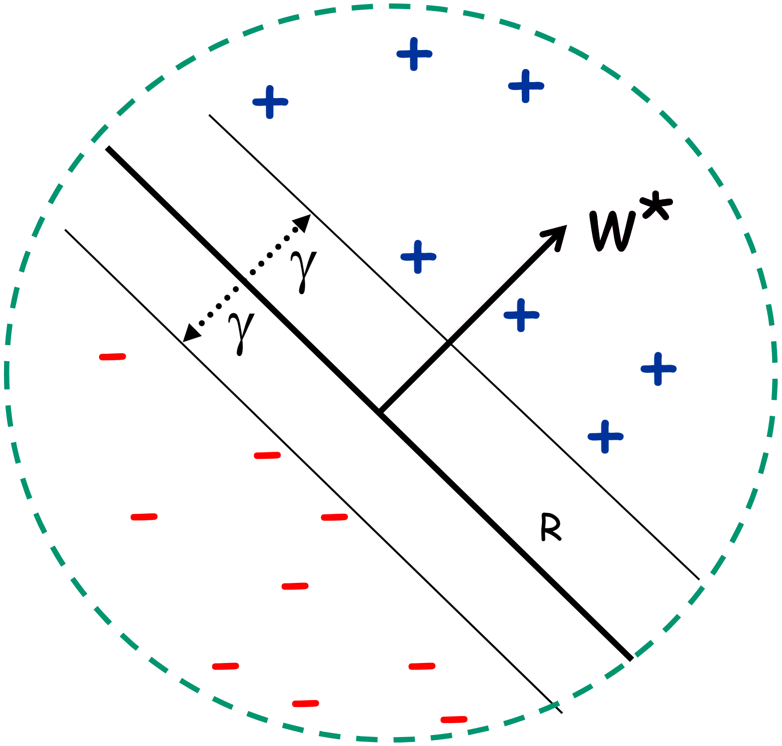


Perceptron: Mistake Bound

Theorem: If $\overset{S}{\text{data}}$ has margin γ and all points inside a ball of radius R, then Perceptron makes $\leq (R/\gamma)^2$ mistakes.

(Normalized margin: multiplying all points by 100, or dividing all points by 100, doesn't change the number of mistakes; algo is invariant to scaling.)

$R = \max\limits_{x_i \in S} \|x_i\|$



$w^* \leftarrow \frac{w}{\|w\|}$

$\gamma_i = y_i x_i^T w^*$

Perceptron Algorithm: Analysis

Theorem: If data has margin γ and all points inside a ball of radius R, then Perceptron makes $\leq (R/\gamma)^2$ mistakes.

Update rule:

- Mistake on positive: $w_{t+1} \leftarrow w_t + x$
- Mistake on negative: $w_{t+1} \leftarrow w_t - x$

$\gamma_i = y_i x_i^T w^*$

$w_{t+1} = w_t + yx$

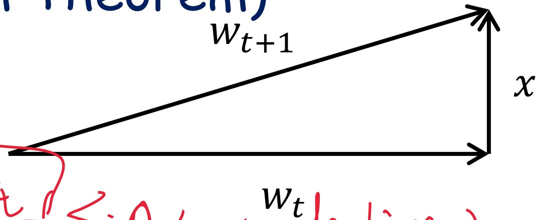
$w_{t+1} \cdot w^* = w_t w^* + \boxed{y x^T w^*} \geq \gamma$

Proof: $R = \max_{x} \|x\|$ $\gamma = \max_{w} \min_{x_i \in S} \gamma_i$

Idea: analyze $w_t \cdot w^*$ and $\|w_t\|$, where w^* is the max-margin sep, $\|w^*\| = 1$.

Claim 1: $w_{t+1} \cdot w^* \geq w_t \cdot w^* + \gamma$. (because $l(x)x \cdot w^* \geq \gamma$)

Claim 2: $\|w_{t+1}\|^2 \leq \|w_t\|^2 + R^2$. (by Pythagorean Theorem)



$\|w_{t+1}\|^2 = \|w_t + yx\|^2$

$= \|w_t\|^2 + \underbrace{\|yx\|^2}_{\leq R^2} + \boxed{2 y x^T w_t} \leq 0$ (prediction mistake)

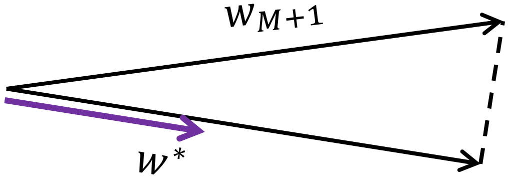
$\leq \|w_t\|^2 + R^2$

After M mistakes:

$w_{M+1} \cdot w^* \geq \gamma M$ (by Claim 1)

$\|w_{M+1}\| \leq R\sqrt{M}$ (by Claim 2)

$w_{M+1} \cdot w^* \leq \|w_{M+1}\|$ (since w^* is unit length)



So, $\gamma M \leq R\sqrt{M}$, so $M \leq \left(\frac{R}{\gamma}\right)^2$.

Perceptron Extensions

- Can use it to find a consistent separator (by cycling through the data).

- One can convert the mistake bound guarantee into a distributional guarantee too (for the case where the x_is come from a fixed distribution).

- Can be adapted to the case where there is no perfect separator as long as the so called hinge loss (i.e., the total distance needed to move the points to classify them correctly large margin) is small.

- Can be kernelized to handle non-linear decision boundaries!

Perceptron Discussion

- Simple online algorithm for learning linear separators with a nice guarantee that depends only on the geometric (aka L_2, L_2) margin.

- It can be kernelized to handle non-linear decision boundaries --- see next class!

- Simple, but very useful in applications like Branch prediction; it also has interesting extensions to structured prediction.