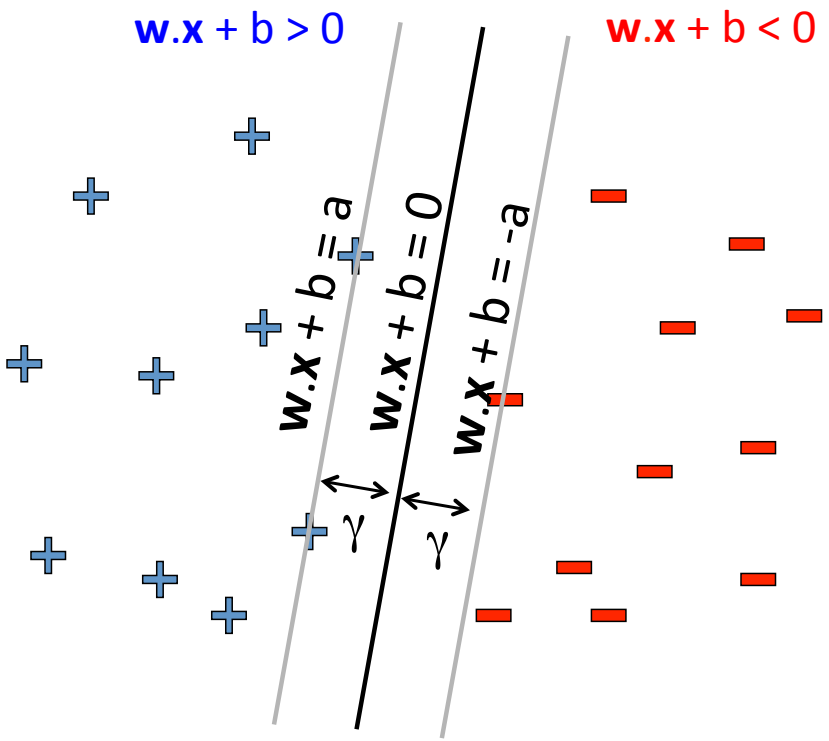
10-601 Recitation

William Bishop

Agenda

- Support Vector Machines
- Boosting

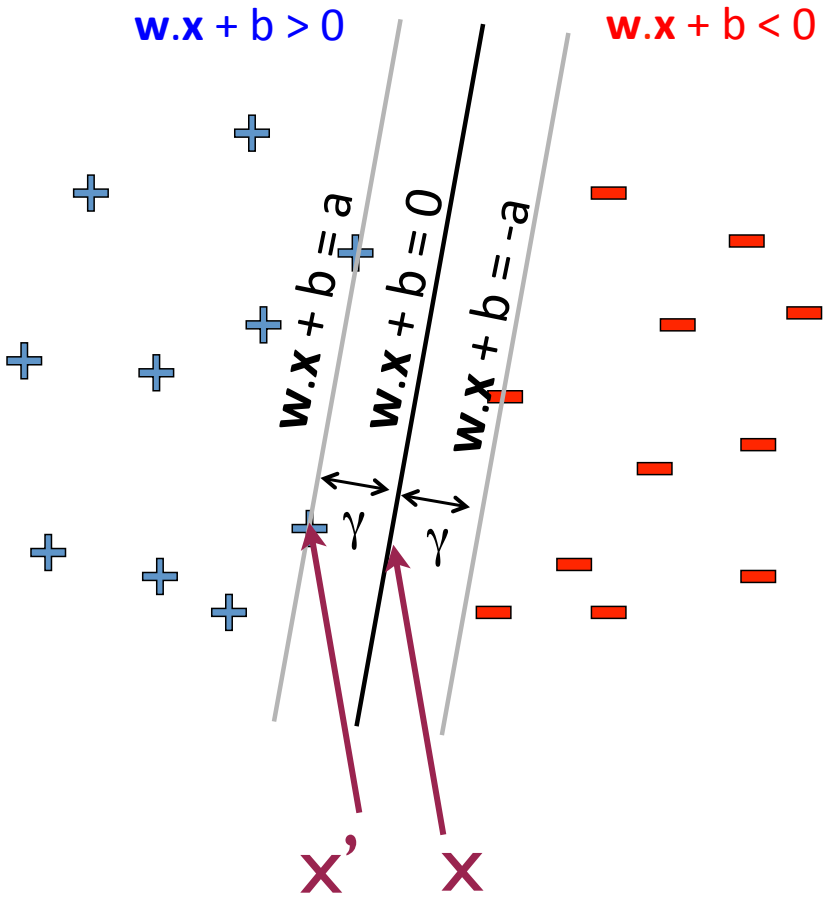
Support Vector Machines



Would like to express the margin a as a function of w and a.

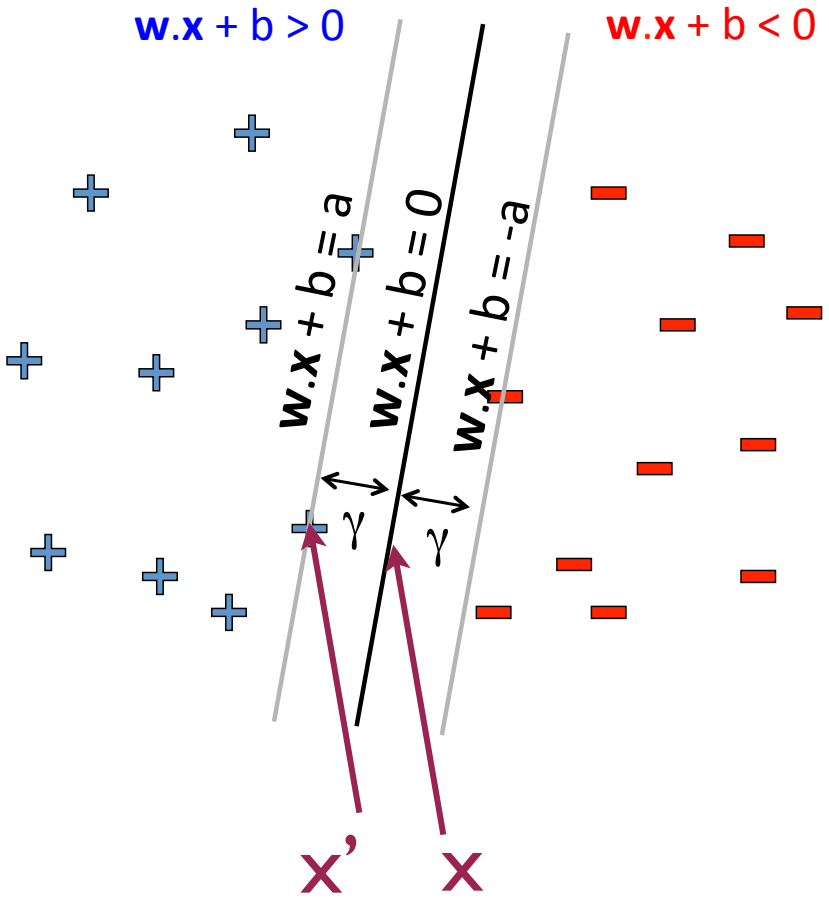
Slide from lecture.

Support Vector Machines



$$(\mathbf{x}' - \mathbf{x})^T \left(\frac{\mathbf{w}}{||\mathbf{w}||} \right) = \gamma$$

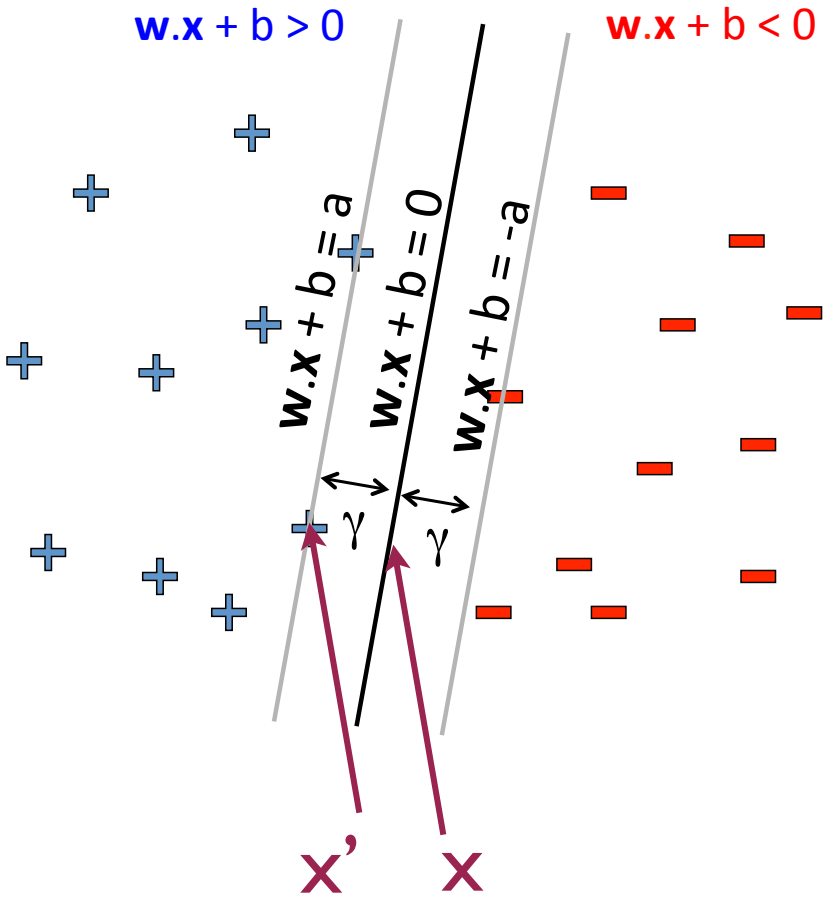
Support Vector Machines



$$(\mathbf{x}' - \mathbf{x})^T \left(\frac{\mathbf{w}}{||\mathbf{w}||} \right) = \gamma$$

$$(\mathbf{x}'^T \mathbf{w} - \mathbf{x}^T \mathbf{w}) \left(\frac{1}{||\mathbf{w}||} \right) = \gamma$$

Support Vector Machines

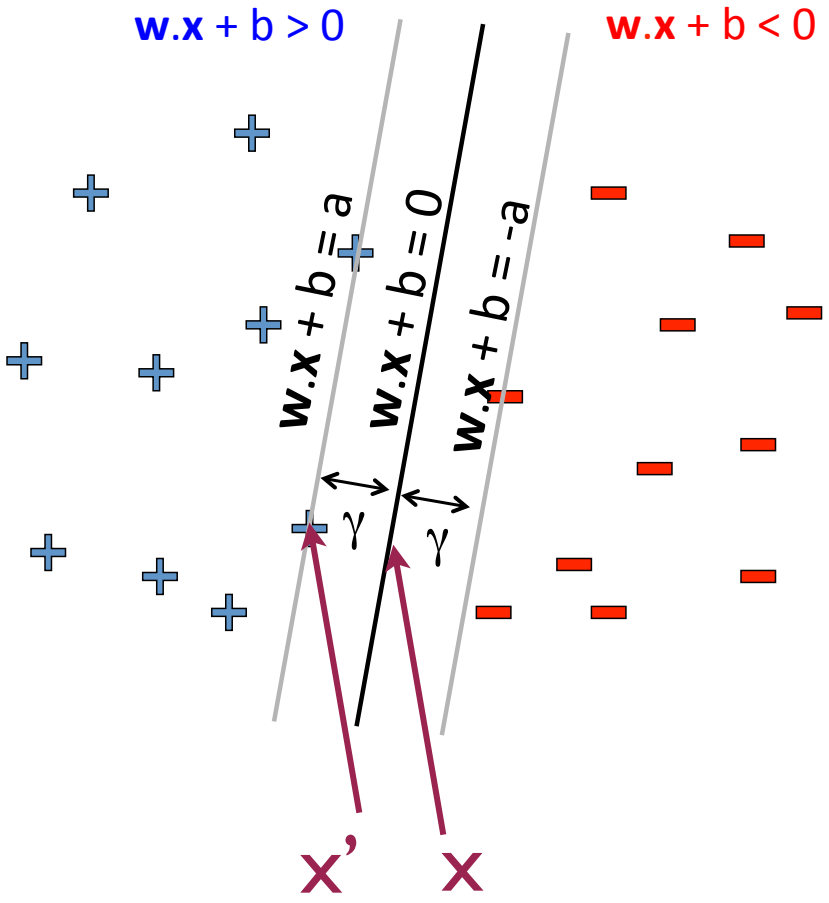


$$(\mathbf{x}' - \mathbf{x})^T \left(\frac{\mathbf{w}}{||\mathbf{w}||} \right) = \gamma$$

$$(\mathbf{x}'^T \mathbf{w} - \mathbf{x}^T \mathbf{w}) \left(\frac{1}{||\mathbf{w}||} \right) = \gamma$$

$$(\mathbf{x}'^T \mathbf{w} - \mathbf{x}^T \mathbf{w} - b + b) \left(\frac{1}{||\mathbf{w}||} \right) = \gamma$$

Support Vector Machines



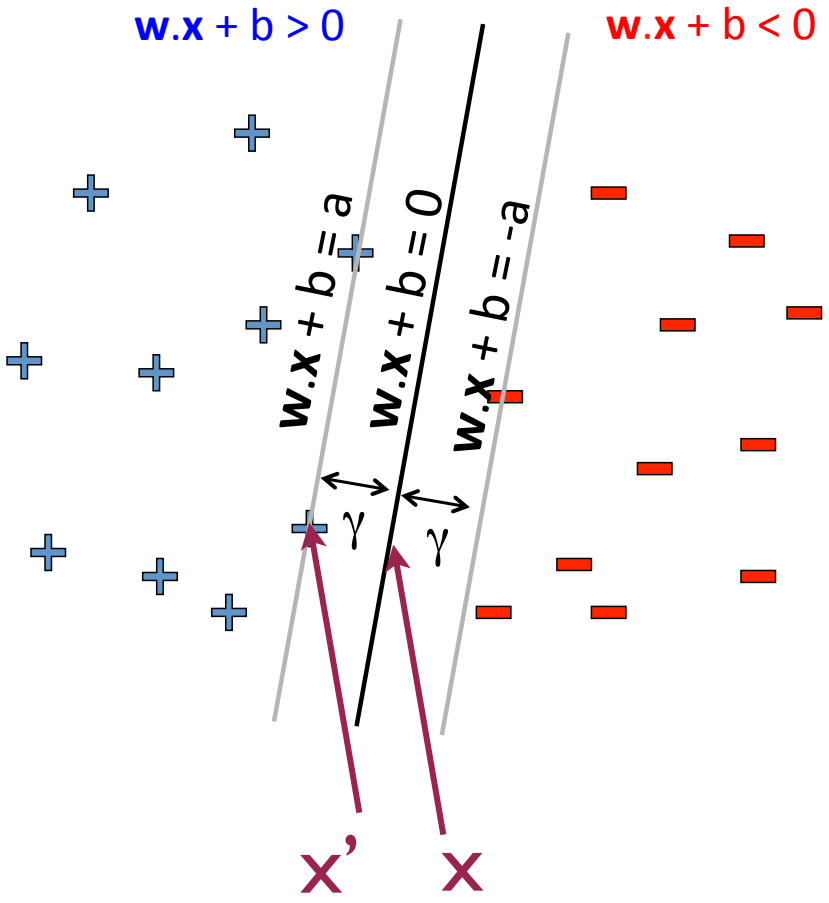
$$(\mathbf{x}' - \mathbf{x})^T \left(\frac{\mathbf{w}}{||\mathbf{w}||} \right) = \gamma$$

$$(\mathbf{x}'^T \mathbf{w} - \mathbf{x}^T \mathbf{w}) \left(\frac{1}{||\mathbf{w}||} \right) = \gamma$$

$$(\mathbf{x}'^T \mathbf{w} - \mathbf{x}^T \mathbf{w} - b + b) \left(\frac{1}{||\mathbf{w}||} \right) = \gamma$$

$$([\mathbf{x}'^T \mathbf{w} + b] - [(\mathbf{x}^T \mathbf{w} + b]) \left(\frac{1}{||\mathbf{w}||} \right) = \gamma$$

Support Vector Machines



$$(\mathbf{x}' - \mathbf{x})^T \left(\frac{\mathbf{w}}{||\mathbf{w}||} \right) = \gamma$$

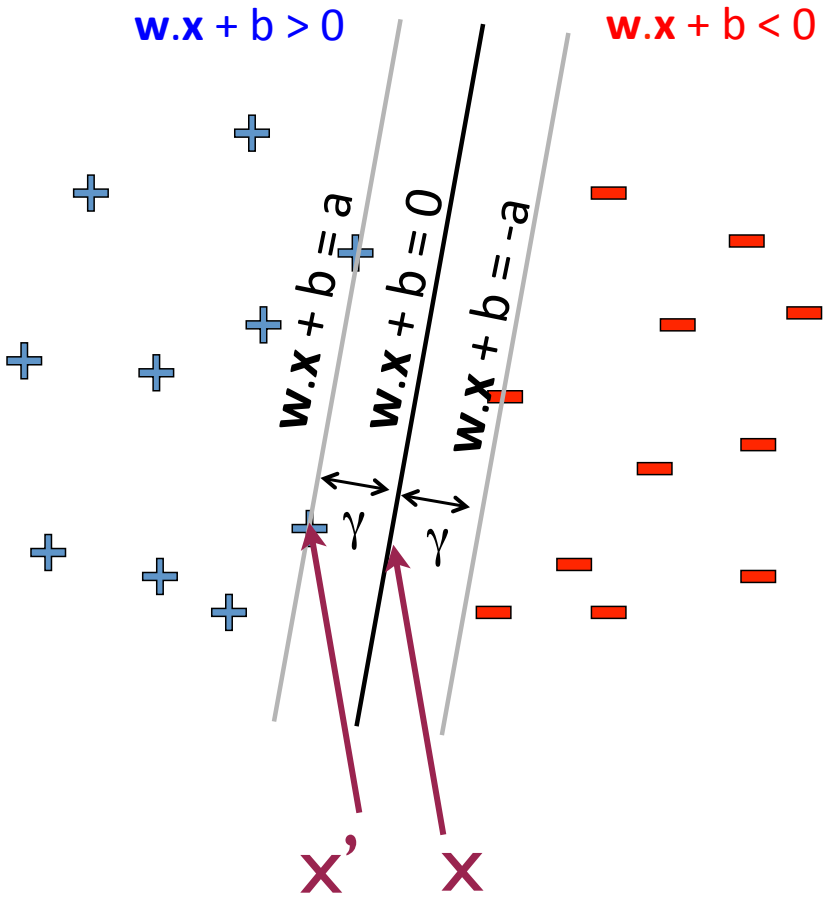
$$(\mathbf{x}'^T \mathbf{w} - \mathbf{x}^T \mathbf{w}) \left(\frac{1}{||\mathbf{w}||} \right) = \gamma$$

$$(\mathbf{x}'^T \mathbf{w} - \mathbf{x}^T \mathbf{w} - b + b) \left(\frac{1}{||\mathbf{w}||} \right) = \gamma$$

$$([\mathbf{x}'^T \mathbf{w} + b] - [(\mathbf{x}^T \mathbf{w} + b]) \left(\frac{1}{||\mathbf{w}||} \right) = \gamma$$

$$([a] - [0]) \left(\frac{1}{||\mathbf{w}||} \right) = \gamma$$

Support Vector Machines



$$(\mathbf{x}' - \mathbf{x})^T \left(\frac{\mathbf{w}}{||\mathbf{w}||}\right) = \gamma$$

$$(\mathbf{x}'^T\mathbf{w} - \mathbf{x}^T\mathbf{w}) \left(\frac{1}{||\mathbf{w}||}\right) = \gamma$$

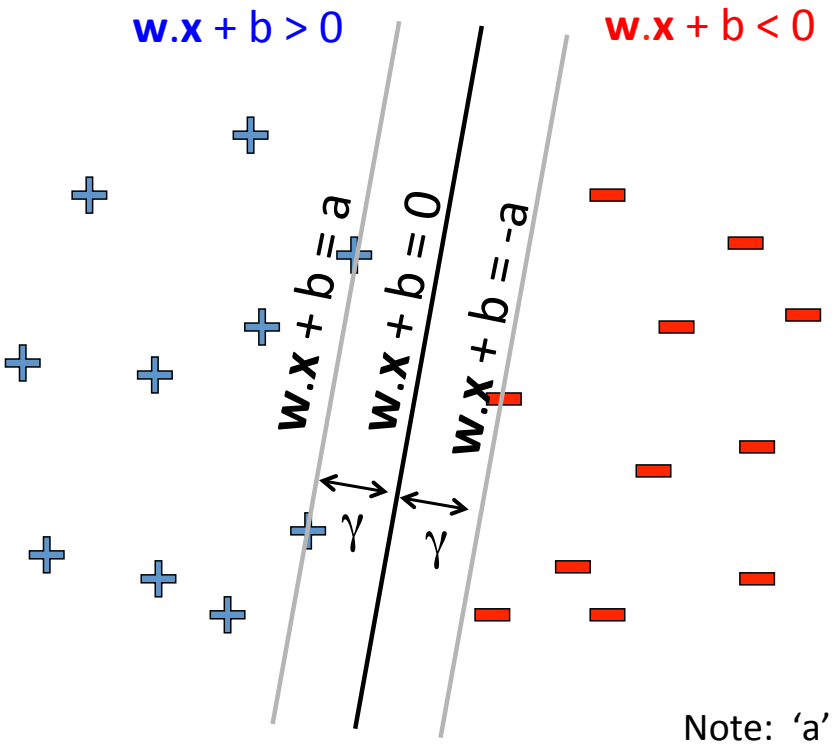
$$(\mathbf{x}'^T\mathbf{w} - \mathbf{x}^T\mathbf{w} - b + b) \left(\frac{1}{||\mathbf{w}||}\right) = \gamma$$

$$([\mathbf{x}'^T\mathbf{w} + b] - [(\mathbf{x}^T\mathbf{w} + b]) \left(\frac{1}{||\mathbf{w}||}\right) = \gamma$$

$$([a] - [0]) \left(\frac{1}{||\mathbf{w}||}\right) = \gamma$$

$$\frac{a}{||\mathbf{w}||} = \gamma$$

Support Vector Machines



Distance of closest examples from the line/hyperplane

$$\text{margin} = \gamma = a/\|w\|$$

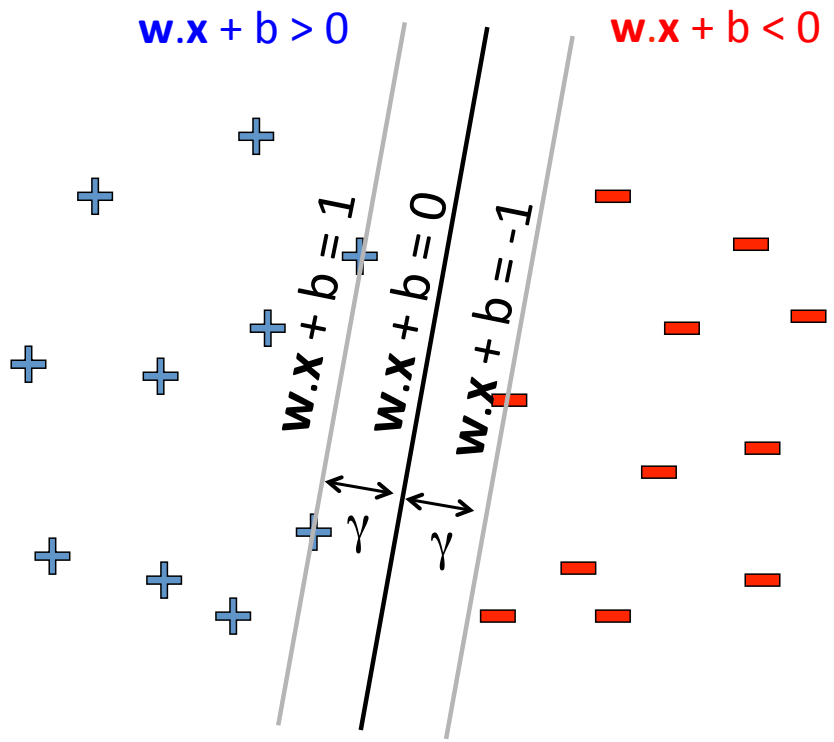
$$\max_{\mathbf{w},b} \ \gamma = a/\|w\|$$

$$\text{s.t. } (\mathbf{w}.\mathbf{x}_j + b)\, y_j \geq a \quad \forall j$$

Note: 'a' is arbitrary (can normalize equations by a)

Slide from lecture.

Support Vector Machines



$$\max_{\mathbf{w},b} \gamma = 1/\|\mathbf{w}\|$$
$$\text{s.t. } (\mathbf{w}.\mathbf{x}_j + b)\, y_j \geq 1 \quad \forall j$$

$$\min_{\mathbf{w},b} \mathbf{w}.\mathbf{w}$$
$$\text{s.t. } (\mathbf{w}.\mathbf{x}_j + b)\, y_j \geq 1 \quad \forall j$$

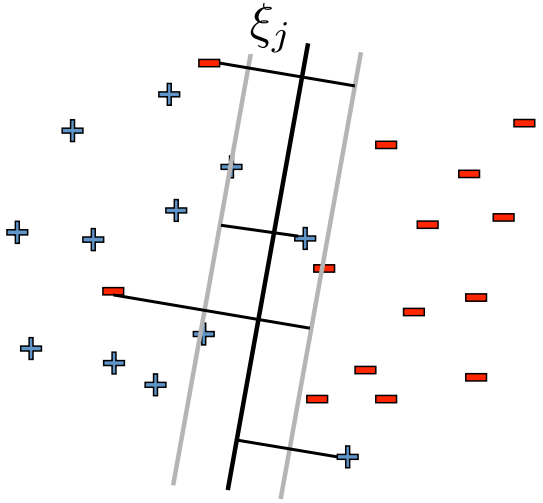
Solve efficiently by quadratic programming (QP)

- Well-studied solution algorithms

Slide from lecture.

Support Vector Machines

Allow "error" in classification



Soft margin approach

$$\min_{\mathbf{w},b,\xi} \ \mathbf{w}.\mathbf{w} + C \sum_j \xi_j$$

$$\text{s.t. } (\mathbf{w}.\mathbf{x}_j + b)\, y_j \geq 1 - \xi_j \quad \forall j$$

$$\xi_j \geq 0 \quad \forall j$$

ξ_j - "slack" variables
= (>1 if x_j misclassifed)
pay linear penalty if mistake

C - tradeoff parameter (chosen by cross-validation)

Still QP ☺

Slide from lecture.

Support Vector Machines

The Primal Problem for the Linearly Separable Case:

$$\min_{\mathbf{w},b} \left(\mathbf{w}^T\mathbf{w}\right)$$

$$\text{s.t. } (\mathbf{w}^T\mathbf{x}_j + b)y_j \geq 1 \qquad \forall j$$

$$L(\mathbf{w}, b, \alpha_j) =$$

Support Vector Machines

The Primal Problem for the Linearly Separable Case:

$$\min_{\mathbf{w},b} \left(\mathbf{w}^T\mathbf{w}\right)$$

$$\text{s.t. } (\mathbf{w}^T\mathbf{x}_j + b)y_j \geq 1 \qquad \forall j$$

$$L(\mathbf{w}, b, \alpha_j) = \mathbf{w}^T\mathbf{w} -$$

Support Vector Machines

The Primal Problem for the Linearly Separable Case:

$$\min_{\mathbf{w},b} \left(\mathbf{w}^T\mathbf{w}\right)$$

$$\text{s.t. } (\mathbf{w}^T\mathbf{x}_j + b)y_j \geq 1 \qquad \forall j$$

$$L(\mathbf{w}, b, \alpha_j) = \mathbf{w}^T\mathbf{w} - \sum_j \alpha_j \left(\mathbf{w}^T\mathbf{x}_j + b)y_j - 1\right)$$

Support Vector Machines

$$L(\mathbf{w}, b, \alpha_j) = \mathbf{w}^T\mathbf{w} - \sum_j \alpha_j \left(\mathbf{w}^T\mathbf{x}_j + b)y_j - 1\right)$$

The Primal Problem: $\min_{\mathbf{w},b} \max_{\alpha_j} [L(\mathbf{w}, b, \alpha_j)]$

$$\text{s.t.} \alpha_j \geq 0 \quad \forall j$$

The Dual Problem: $\max_{\alpha_j} \min_{\mathbf{w},b} [L(\mathbf{w}, b, \alpha_j)]$

$$\text{s.t.} \alpha_j \geq 0 \quad \forall j$$

Support Vector Machines

Solving the dual: $\max_{\alpha_j} \min_{\mathbf{w},b} [L(\mathbf{w}, b, \alpha_j)]$

$$\text{s.t.} \alpha_j \geq 0 \quad \forall j$$

$$L(\mathbf{w}, b, \alpha_j) = \frac{1}{2}\mathbf{w}^T\mathbf{w} - \sum_j \alpha_j \left(\mathbf{w}^T\mathbf{x}_j + b)y_j - 1\right)$$

$$\frac{\partial L}{\partial \mathbf{w}} =$$

$$\frac{\partial L}{\partial b} =$$

Support Vector Machines

Solving the dual: $\max_{\alpha_j} \min_{\mathbf{w},b} [L(\mathbf{w}, b, \alpha_j)]$

$$\text{s.t.} \alpha_j \geq 0 \quad \forall j$$

$$L(\mathbf{w}, b, \alpha_j) = \frac{1}{2}\mathbf{w}^T\mathbf{w} - \sum_j \alpha_j \left(\mathbf{w}^T\mathbf{x}_j + b)y_j - 1\right)$$

$$\frac{\partial L}{\partial \mathbf{w}} = \mathbf{w} - \sum_j \alpha_j \mathbf{x}_j y_j \qquad \mathbf{w} = \sum_j \alpha_j \mathbf{x}_j y_j$$

$$\frac{\partial L}{\partial b} =$$

Support Vector Machines

Solving the dual:

$$\max_{\alpha_j} \min_{\mathbf{w},b} \left[L(\mathbf{w}, b, \alpha_j) \right]$$

$$\text{s.t.} \alpha_j \geq 0 \quad \forall j$$

$$L(\mathbf{w}, b, \alpha_j) = \frac{1}{2}\mathbf{w}^T\mathbf{w} - \sum_j \alpha_j \left(\mathbf{w}^T\mathbf{x}_j + b)y_j - 1 \right)$$

$$\frac{\partial L}{\partial \mathbf{w}} = \mathbf{w} - \sum_j \alpha_j \mathbf{x}_j y_j \qquad \mathbf{w} = \sum_j \alpha_j \mathbf{x}_j y_j$$

$$\frac{\partial L}{\partial b} = \sum_j \alpha_j y_j \qquad 0 = \sum_j \alpha_j y_j$$

Support Vector Machines

Solving the dual: $$\max_{\alpha_j} \min_{\mathbf{w},b} \left[L(\mathbf{w}, b, \alpha_j)\right]$$

$$\text{s.t.} \alpha_j \geq 0 \quad \forall j$$

$$\frac{1}{2}\mathbf{w}^T\mathbf{w} - \sum_j \alpha_j \left(\mathbf{w}^T\mathbf{x}_j + b)y_j - 1\right)$$

Support Vector Machines

Solving the dual: $$\max_{\alpha_j} \min_{\mathbf{w},b} \left[L(\mathbf{w}, b, \alpha_j)\right]$$

$$\text{s.t.} \alpha_j \geq 0 \quad \forall j$$

$$\frac{1}{2}\mathbf{w}^T\mathbf{w} - \sum_j \alpha_j \left(\mathbf{w}^T\mathbf{x}_j + b)y_j - 1\right)$$

$$= \frac{1}{2}\mathbf{w}^T\mathbf{w} - \sum_j \alpha_j \mathbf{w}^T\mathbf{x}_j y_j - b\sum_j \alpha_j y_j + \sum_j \alpha_j$$

Support Vector Machines

Solving the dual: $$\max_{\alpha_j} \min_{\mathbf{w},b} \left[L(\mathbf{w}, b, \alpha_j)\right]$$

$$\text{s.t.} \alpha_j \geq 0 \quad \forall j$$

$$\frac{1}{2}\mathbf{w}^T\mathbf{w} - \sum_j \alpha_j \left(\mathbf{w}^T\mathbf{x}_j + b)y_j - 1\right)$$

$$= \frac{1}{2}\mathbf{w}^T\mathbf{w} - \sum_j \alpha_j \mathbf{w}^T\mathbf{x}_j y_j - b\sum_j \alpha_j y_j + \sum_j \alpha_j$$

$$= \frac{1}{2}\left(\sum_j \alpha_j \mathbf{x}_j y_j\right)^T \left(\sum_i \alpha_i \mathbf{x}_i y_i\right) - \sum_j \alpha_j \left(\sum_i \alpha_i \mathbf{x}_i y_i\right)^T \mathbf{x}_j y_j + \sum_j \alpha_j$$

Support Vector Machines

Solving the dual: $$\max_{\alpha_j} \min_{\mathbf{w},b} \left[L(\mathbf{w}, b, \alpha_j)\right]$$

$$\text{s.t.} \alpha_j \geq 0 \quad \forall j$$

$$\frac{1}{2}\mathbf{w}^T\mathbf{w} - \sum_j \alpha_j \left(\mathbf{w}^T\mathbf{x}_j + b)y_j - 1\right)$$

$$= \frac{1}{2}\mathbf{w}^T\mathbf{w} - \sum_j \alpha_j \mathbf{w}^T\mathbf{x}_j y_j - b\sum_j \alpha_j y_j + \sum_j \alpha_j$$

$$= \frac{1}{2}\left(\sum_j \alpha_j \mathbf{x}_j y_j\right)^T \left(\sum_i \alpha_i \mathbf{x}_i y_i\right) - \sum_j \alpha_j \left(\sum_i \alpha_i \mathbf{x}_i y_i\right)^T \mathbf{x}_j y_j + \sum_j \alpha_j$$

$$= \sum_j \alpha_j - \frac{1}{2}\sum_i \sum_j \alpha_i \alpha_j y_i y_j \mathbf{x}_i^T \mathbf{x}_j$$

Support Vector Machines

Solving the dual:

$$\sum_j \alpha_j - \frac{1}{2} \sum_i \sum_j \alpha_i \alpha_j y_i y_j \mathbf{x}_i^T \mathbf{x}_j$$

Maximize this with respect to α_j.

Then:

$$\mathbf{w} = \sum_j \alpha_j \mathbf{x}_j y_j$$

$$b = y_j - \mathbf{w}^T \mathbf{x}_j$$

for any j where $\alpha_j > 0$

Support Vector Machines

So why work with the dual?

$$\max_{\alpha_j} \left(\sum_j \alpha_j - \frac{1}{2} \sum_i \sum_j \alpha_i \alpha_j y_i y_j \mathbf{x}_i^T \mathbf{x}_j \right)$$

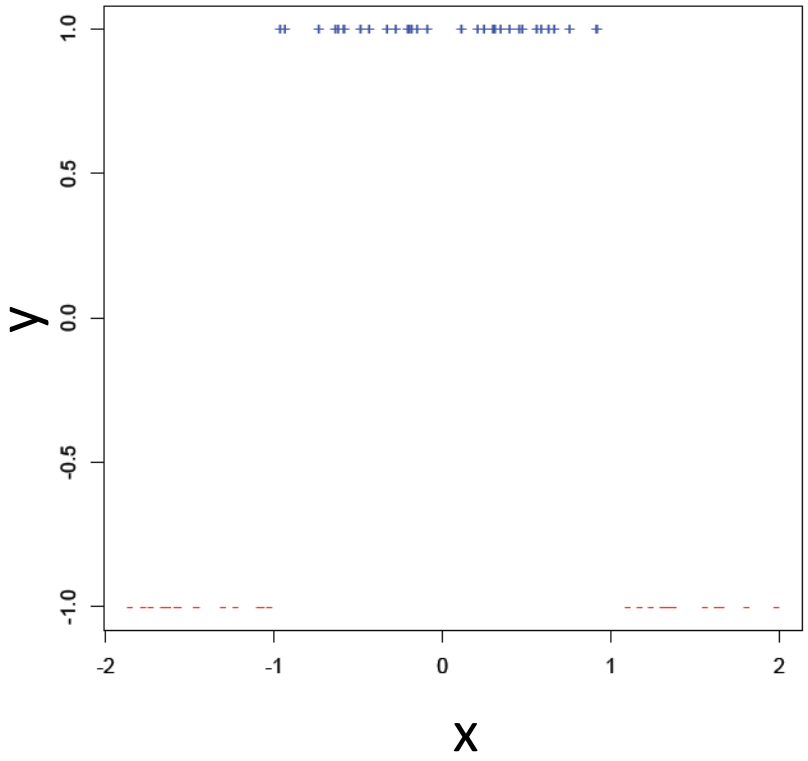
$$\text{s.t.} \alpha_j \geq 0$$

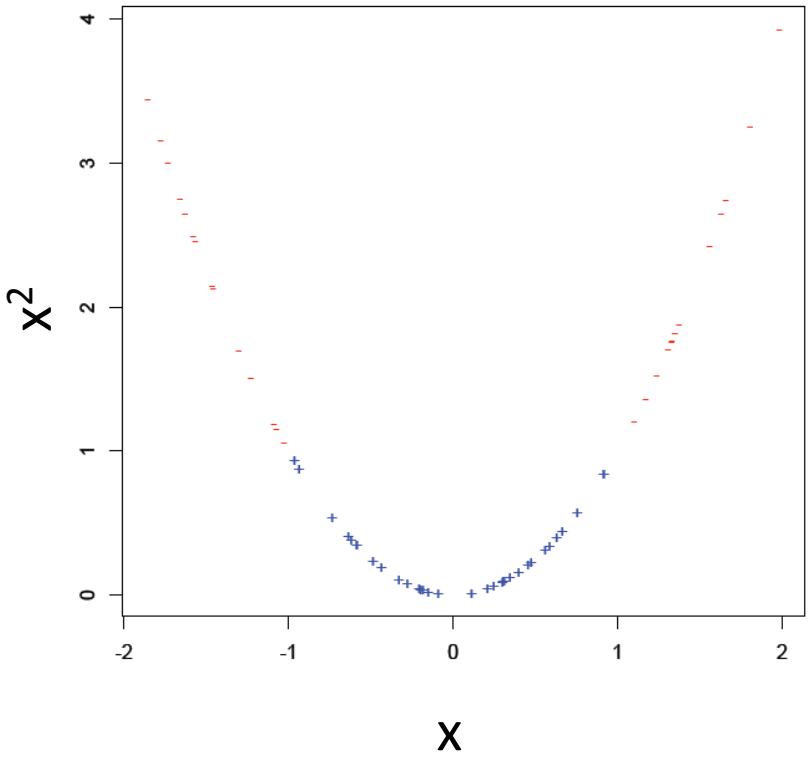
$$\sum_j \alpha_j y_j = 0$$

Just a dot product!

Support Vector Machines

Using non-linear features to get linear separation

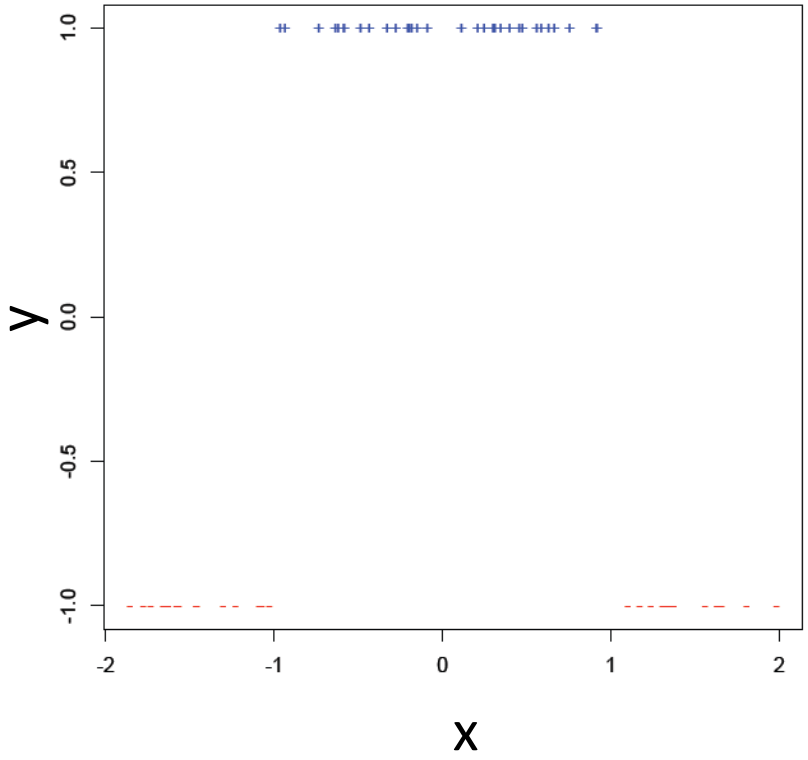


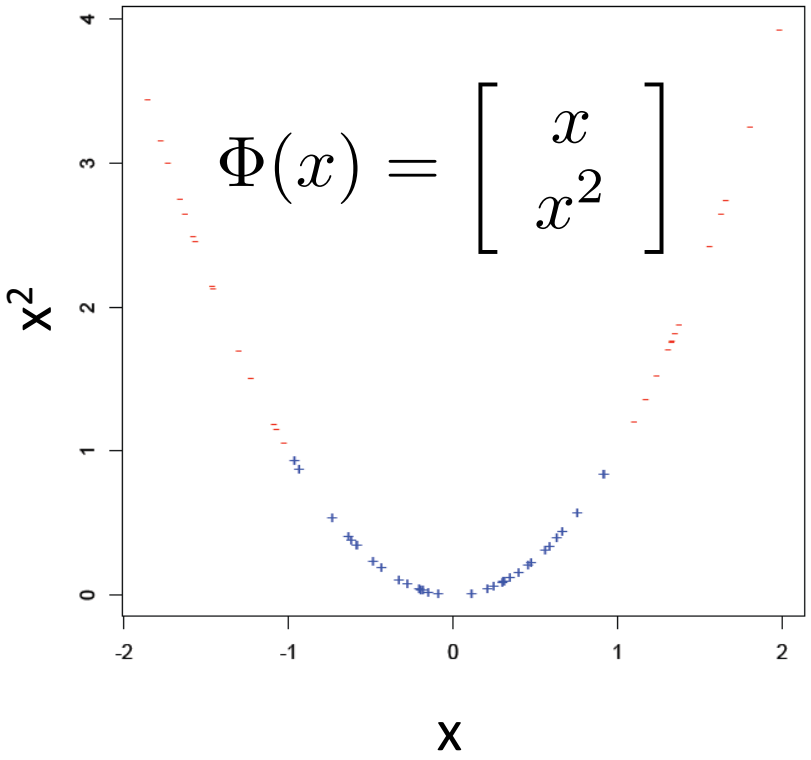


From class slides.

Support Vector Machines

Using non-linear features to get linear separation





From class slides.

Support Vector Machines

So why work with the dual?

$$\max_{\alpha_j} \left(\sum_j \alpha_j - \frac{1}{2} \sum_i \sum_j \alpha_i \alpha_j y_i y_j \Phi(\mathbf{x}_i)^T \Phi(\mathbf{x}_j) \right)$$

$$\text{s.t.} \alpha_j \geq 0$$

$$\sum_j \alpha_j y_j = 0$$

Just a dot product!

Support Vector Machines

So why work with the dual?

$$\max_{\alpha_j} \left(\sum_j \alpha_j - \frac{1}{2} \sum_i \sum_j \alpha_i \alpha_j y_i y_j K(\mathbf{x}_i, \mathbf{x}_j) \right)$$

$$\text{s.t.} \alpha_j \geq 0$$

$$\sum_j \alpha_j y_j = 0$$

$$K(x_i, x_j) = \Phi(x_i)^T \Phi(x_j) \longleftarrow$$

Kernel function keeps us from having to work in a high dimensional space.

Support Vector Machines

$$\mathbf{x} = \begin{bmatrix} x_1 \\ x_2 \end{bmatrix} \quad \mathbf{z} = \begin{bmatrix} z_1 \\ z_2 \end{bmatrix} \quad \Phi(\mathbf{x}) = \text{polynomials of degree exactly } m$$

m=1 $\Phi(\mathbf{x}) \cdot \Phi(\mathbf{z}) = \begin{bmatrix} x_1 \\ x_2 \end{bmatrix} \cdot \begin{bmatrix} z_1 \\ z_2 \end{bmatrix} = x_1 z_1 + x_2 z_2 = \mathbf{x} \cdot \mathbf{z}$

m=2 $\Phi(\mathbf{x}) \cdot \Phi(\mathbf{z}) = \begin{bmatrix} x_1^2 \\ \sqrt{2} x_1 x_2 \\ x_2^2 \end{bmatrix} \cdot \begin{bmatrix} z_1^2 \\ \sqrt{2} z_1 z_2 \\ z_2^2 \end{bmatrix} = x_1^2 z_1^2 + x_2^2 z_2^2 + 2 x_1 x_2 z_1 z_2$

$$= (x_1 z_1 + x_2 z_2)^2$$

$$= (\mathbf{x} \cdot \mathbf{z})^2$$

m $\Phi(\mathbf{x}) \cdot \Phi(\mathbf{z}) = K(\mathbf{x}, \mathbf{z}) = (\mathbf{x} \cdot \mathbf{z})^m = K(\mathbf{x},\mathbf{z})$

Don't store high-dim features - Only evaluate dot-products with kernels

From class slides.

Support Vector Machines

How about classification?

$$\mathbf{w} =$$

Support Vector Machines

How about classification?

$$\mathbf{w} = \sum_j \alpha_j y_j \Phi(\mathbf{x}_j)$$

Support Vector Machines

How about classification?

$$\mathbf{w} = \sum_j \alpha_j y_j \Phi(\mathbf{x}_j)$$

$$\Phi(\mathbf{u})^T \mathbf{w} = \Phi(\mathbf{u})^T \sum_j \alpha_j y_j \Phi(\mathbf{x}_j)$$

Support Vector Machines

How about classification?

$$\mathbf{w} = \sum_j \alpha_j y_j \Phi(\mathbf{x}_j)$$

$$\Phi(\mathbf{u})^T \mathbf{w} = \Phi(\mathbf{u})^T \sum_j \alpha_j y_j \Phi(\mathbf{x}_j)$$

$$= \sum_j \alpha_j y_j \Phi(\mathbf{u})^T \Phi(\mathbf{x}_j)$$

Support Vector Machines

How about classification?

$$\mathbf{w} = \sum_j \alpha_j y_j \Phi(\mathbf{x}_j)$$

$$\begin{aligned}\Phi(\mathbf{u})^T \mathbf{w} &= \Phi(\mathbf{u})^T \sum_j \alpha_j y_j \Phi(\mathbf{x}_j) \\ &= \sum_j \alpha_j y_j \Phi(\mathbf{u})^T \Phi(\mathbf{x}_j) \\ &= \sum_j \alpha_j y_j K(\mathbf{u}, \mathbf{x}_j)\end{aligned}$$

Support Vector Machines

How about classification?

$$\mathbf{w} = \sum_j \alpha_j y_j \Phi(\mathbf{x}_j) \qquad b =$$

$$\Phi(\mathbf{u})^T \mathbf{w} = \Phi(\mathbf{u})^T \sum_j \alpha_j y_j \Phi(\mathbf{x}_j)$$

$$= \sum_j \alpha_j y_j \Phi(\mathbf{u})^T \Phi(\mathbf{x}_j)$$

$$= \sum_j \alpha_j y_j K(\mathbf{u}, \mathbf{x}_j)$$

Support Vector Machines

How about classification?

$$\mathbf{w} = \sum_j \alpha_j y_j \Phi(\mathbf{x}_j)$$

$$\begin{aligned} b &= y_i - \Phi(\mathbf{x}_i)^T \mathbf{w} \\ &= y_i - \Phi(\mathbf{x}_i)^T \sum_j \alpha_j y_j \Phi(\mathbf{x}_j) \end{aligned}$$

$$\begin{aligned} \Phi(\mathbf{u})^T \mathbf{w} &= \Phi(\mathbf{u})^T \sum_j \alpha_j y_j \Phi(\mathbf{x}_j) \\ &= \sum_j \alpha_j y_j \Phi(\mathbf{u})^T \Phi(\mathbf{x}_j) \\ &= \sum_j \alpha_j y_j K(\mathbf{u}, \mathbf{x}_j) \end{aligned}$$

Support Vector Machines

How about classification?

$$\mathbf{w} = \sum_j \alpha_j y_j \Phi(\mathbf{x}_j)$$

$$\Phi(\mathbf{u})^T \mathbf{w} = \Phi(\mathbf{u})^T \sum_j \alpha_j y_j \Phi(\mathbf{x}_j)$$

$$= \sum_j \alpha_j y_j \Phi(\mathbf{u})^T \Phi(\mathbf{x}_j)$$

$$= \sum_j \alpha_j y_j K(\mathbf{u}, \mathbf{x}_j)$$

$$b = y_i - \Phi(\mathbf{x}_i)^T \mathbf{w}$$

$$= y_i - \Phi(\mathbf{x}_i)^T \sum_j \alpha_j y_j \Phi(\mathbf{x}_j)$$

Support Vector Machines

How about classification?

$$\mathbf{w} = \sum_j \alpha_j y_j \Phi(\mathbf{x}_j)$$

$$\begin{aligned}\Phi(\mathbf{u})^T \mathbf{w} &= \Phi(\mathbf{u})^T \sum_j \alpha_j y_j \Phi(\mathbf{x}_j) \\ &= \sum_j \alpha_j y_j \Phi(\mathbf{u})^T \Phi(\mathbf{x}_j) \\ &= \sum_j \alpha_j y_j K(\mathbf{u}, \mathbf{x}_j)\end{aligned}$$

$$\begin{aligned}b &= y_i - \Phi(\mathbf{x}_i)^T \mathbf{w} \\ &= y_i - \Phi(\mathbf{x}_i)^T \sum_j \alpha_j y_j \Phi(\mathbf{x}_j) \\ &= y_i - \sum_j \alpha_j y_j \Phi(\mathbf{x}_i)^T \Phi(\mathbf{x}_j)\end{aligned}$$

Support Vector Machines

How about classification?

$$\mathbf{w} = \sum_j \alpha_j y_j \Phi(\mathbf{x}_j)$$

$$\begin{aligned}\Phi(\mathbf{u})^T \mathbf{w} &= \Phi(\mathbf{u})^T \sum_j \alpha_j y_j \Phi(\mathbf{x}_j) \\ &= \sum_j \alpha_j y_j \Phi(\mathbf{u})^T \Phi(\mathbf{x}_j) \\ &= \sum_j \alpha_j y_j K(\mathbf{u}, \mathbf{x}_j)\end{aligned}$$

$$\begin{aligned}b &= y_i - \Phi(\mathbf{x}_i)^T \mathbf{w} \\ &= y_i - \Phi(\mathbf{x}_i)^T \sum_j \alpha_j y_j \Phi(\mathbf{x}_j) \\ &= y_i - \sum_j \alpha_j y_j \Phi(\mathbf{x}_i)^T \Phi(\mathbf{x}_j) \\ &= y_i - \sum_j \alpha_j y_j K(\mathbf{x}_i, \mathbf{x}_j)\end{aligned}$$

for any i where $\alpha_j > 0$

Support Vector Machines

Dual SVM – non-separable case

$$\text{maximize}_{\alpha} \quad \sum_i \alpha_i - \frac{1}{2} \sum_{i,j} \alpha_i \alpha_j y_i y_j \mathbf{x}_i . \mathbf{x}_j$$

$$\sum_i \alpha_i y_i = 0$$

$$C \geq \alpha_i \geq 0$$

comes from $\frac{\partial L}{\partial \mu} = 0$

Intuition:

Earlier - If constraint violated, $\alpha_i \rightarrow \infty$

Now - If constraint violated, $\alpha_i \leq C$

Dual problem is also QP → Solution gives α_js

$$\mathbf{w} = \sum_i \alpha_i y_i \mathbf{x}_i$$

$$b = y_k - \mathbf{w} . \mathbf{x}_k$$

for any k where $C > \alpha_k > 0$

From class slides.

Boosting

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

From class slides.

Boosting

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

All comes down to how we pick h_t and α_t.

Boosting

The algorithm:

Given $(x_1, y_1), \ldots, (x_m, y_m)$

1. Initialize $D_1(i) = 1/m$
2. For $t = 1, \ldots, T$:
   (a) Train a weak classifier using $(x_1, y_1), \ldots, (x_m, y_m)$ and D_t
   (b) Choose α_t.
   (c) Update weights and from D_{t+1}.

By virtue of changing weights, you get different classifiers.

Final Classifier:

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

Boosting

So now the question is how do we pick α_t and D_{t+1} intelligently?

Boosting

So now the question is how do we pick α_t and D_{t+1} intelligently?

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

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

Boosting

So now the question is how do we pick α_t and D_{t+1} intelligently?

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

$$\epsilon_t = \sum_{i=1}^{m} D_t(i) \delta \left(h_t(x_i) \neq y_i \right)$$

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

Boosting

So now the question is how do we pick α_t and D_{t+1} intelligently?

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

$$\epsilon_t = \sum_{i=1}^{m} D_t(i) \delta \left(h_t(x_i) \neq y_i \right)$$

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

ε_t = 0 if h_t perfectly classifies all weighted data pts — $\alpha_t = \infty$

ε_t = 1 if h_t perfectly wrong => $-h_t$ perfectly right — $\alpha_t = -\infty$

ε_t = 0.5 — $\alpha_t = 0$

From class slides.

Boosting

So why are these good choices for α_t and D_{t+1}?

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

From class slides.

Thus, the goal is to show by picking α_t and D_{t+1}, we get this result.

Boosting

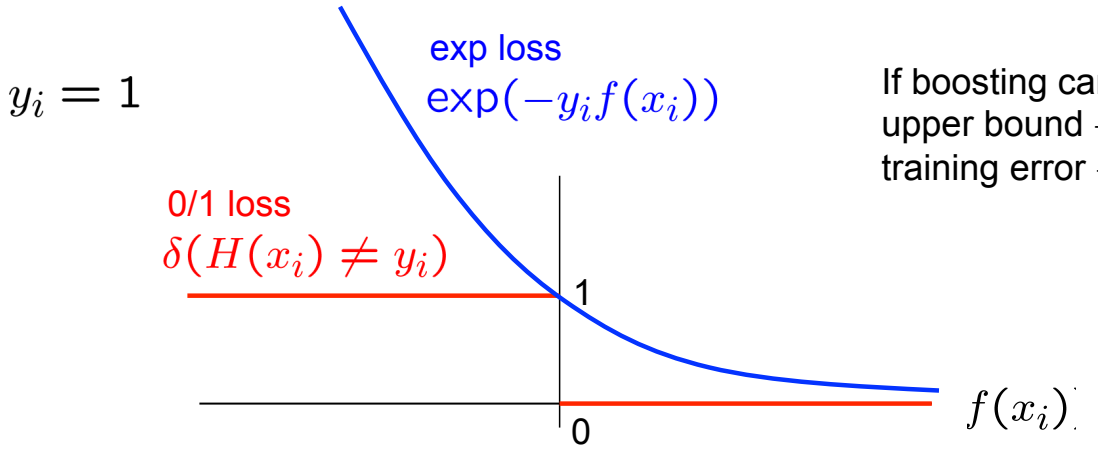
First, let's bound the training error:

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

From class slides.

Boosting

Important:

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

$$D_1(i) = \frac{1}{}$$

Boosting

Important: $$\frac{1}{m}\sum_{i=1}^{m} e^{-y_i f(x_i)} = \prod_t Z_t$$

$$D_1(i) = \frac{1}{m}$$

$$D_2(i) = \frac{D_1(i)e^{-\alpha_1 y_i h_1(x_i)}}{Z_1} = \frac{1}{m}\frac{e^{-\alpha_1 y_i h_1(x_i)}}{Z_1}$$

Boosting

Important:

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

$$D_1(i) = \frac{1}{m}$$

$$D_2(i) = \frac{D_1(i)e^{-\alpha_1 y_i h_1(x_i)}}{Z_1} = \frac{1}{m}\frac{e^{-\alpha_1 y_i h_1(x_i)}}{Z_1}$$

$$D_3(i) = \frac{D_2(i)e^{-\alpha_2 y_i h_2(x_i)}}{Z_2} = \frac{1}{m}\frac{e^{-\alpha_1 y_i h_1(x_i)}e^{-\alpha_2 y_i h_2(x_i)}}{Z_1 Z_2}$$

Boosting

Important:

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

$$D_1(i) = \frac{1}{m}$$

$$D_2(i) = \frac{D_1(i)e^{-\alpha_1 y_i h_1(x_i)}}{Z_1} = \frac{1}{m}\frac{e^{-\alpha_1 y_i h_1(x_i)}}{Z_1}$$

$$D_3(i) = \frac{D_2(i)e^{-\alpha_2 y_i h_2(x_i)}}{Z_2} = \frac{1}{m}\frac{e^{-\alpha_1 y_i h_1(x_i)}e^{-\alpha_2 y_i h_2(x_i)}}{Z_1 Z_2}$$

$$D_{T+1}(i) = \frac{1}{m}\frac{e^{-\alpha_1 y_i h_1(x_i)}\ldots e^{-\alpha_T y_i h_T(x_i)}}{Z_1 \ldots Z_T} = \frac{1}{m}\frac{e^{-y_i \sum_{t=1}^{T}\alpha_t h_t(x_i)}}{\prod_{t=1}^{T} Z_t} = \frac{1}{m}\frac{e^{-y_i f(x_i)}}{\prod_{t=1}^{T} Z_t}$$

Boosting

Important:

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

$$D_1(i) = \frac{1}{m}$$

$$D_2(i) = \frac{D_1(i)e^{-\alpha_1 y_i h_1(x_i)}}{Z_1} = \frac{1}{m}\frac{e^{-\alpha_1 y_i h_1(x_i)}}{Z_1}$$

$$D_3(i) = \frac{D_2(i)e^{-\alpha_2 y_i h_2(x_i)}}{Z_2} = \frac{1}{m}\frac{e^{-\alpha_1 y_i h_1(x_i)}e^{-\alpha_2 y_i h_2(x_i)}}{Z_1 Z_2}$$

$$D_{T+1}(i) = \frac{1}{m}\frac{e^{-\alpha_1 y_i h_1(x_i)}\ldots e^{-\alpha_T y_i h_T(x_i)}}{Z_1 \ldots Z_T} = \frac{1}{m}\frac{e^{-y_i \sum_{t=1}^{T}\alpha_t h_t(x_i)}}{\prod_{t=1}^{T} Z_t} = \frac{1}{m}\frac{e^{-y_i f(x_i)}}{\prod_{t=1}^{T} Z_t}$$

$$1 = \sum_{i=1}^{m} D_{T+1}(i) = \sum_{i=1}^{m}\frac{1}{m}\frac{e^{-y_i f(x_i)}}{\prod_{t=1}^{T} Z_t} = \frac{1}{m\prod_{t=1}^{T} Z_t}\sum_{i=1}^{m} e^{-y_i f(x_i)}$$

Boosting

Important: $$\frac{1}{m}\sum_{i=1}^{m} e^{-y_i f(x_i)} = \prod_t Z_t$$

$$D_1(i) = \frac{1}{m}$$

$$D_2(i) = \frac{D_1(i)e^{-\alpha_1 y_i h_1(x_i)}}{Z_1} = \frac{1}{m}\frac{e^{-\alpha_1 y_i h_1(x_i)}}{Z_1}$$

$$D_3(i) = \frac{D_2(i)e^{-\alpha_2 y_i h_2(x_i)}}{Z_2} = \frac{1}{m}\frac{e^{-\alpha_1 y_i h_1(x_i)}e^{-\alpha_2 y_i h_2(x_i)}}{Z_1 Z_2}$$

$$D_{T+1}(i) = \frac{1}{m}\frac{e^{-\alpha_1 y_i h_1(x_i)}\ldots e^{-\alpha_T y_i h_T(x_i)}}{Z_1 \ldots Z_T} = \frac{1}{m}\frac{e^{-y_i \sum_{t=1}^{T}\alpha_t h_t(x_i)}}{\prod_{t=1}^{T} Z_t} = \frac{1}{m}\frac{e^{-y_i f(x_i)}}{\prod_{t=1}^{T} Z_t}$$

$$1 = \sum_{i=1}^{m} D_{T+1}(i) = \sum_{i=1}^{m}\frac{1}{m}\frac{e^{-y_i f(x_i)}}{\prod_{t=1}^{T} Z_t} = \frac{1}{m\prod_{t=1}^{T} Z_t}\sum_{i=1}^{m} e^{-y_i f(x_i)}$$

$$\prod_{t=1}^{T} Z_t = \frac{1}{m}\sum_{i=1}^{m} e^{-y_i f(x_i)}$$

Boosting

So where are we?

We've established: $\frac{1}{m}\sum_{i=1}^{m}\delta(H(x_i)\neq y_i) \leq \frac{1}{m}\sum_{i=1}^{m} e^{-y_i f(x_i)} = \prod_{t=1}^{T} Z_t$

So if we minimize Z_t we can minimize bound on training error.

This is where we decide on how to pick α_t.

Boosting

So if we minimize Z_t we can minimize bound on training error.

$$Z_t = \sum_{y_i} D_t(i) e^{\alpha_t y_i h_t(x_i)}$$

$$= \sum_{i: y_i \neq h_t(x_i)} D_t(i) e^{\alpha_t} + \sum_{i: y_i = h_t(x_i)} D_t(i) e^{-\alpha_t}$$

All misclassified samples.

All correctly classified samples.

First trick is to break Z_t up into two sums.

Boosting

So if we minimize Z_t we can minimize bound on training error.

$$Z_t = \sum_{i:y_i \neq h_t(x_i)} D_t(i) e^{\alpha_t} + \sum_{i:y_i = h_t(x_i)} D_t(i) e^{-\alpha_t}$$

Boosting

So if we minimize Z_t we can minimize bound on training error.

$$Z_t = \sum_{i:y_i \neq h_t(x_i)} D_t(i) e^{\alpha_t} + \sum_{i:y_i = h_t(x_i)} D_t(i) e^{-\alpha_t}$$

$$\frac{\partial Z_t}{\partial \alpha_t} = \sum_{i:y_i \neq h_t(x_i)} D_t(i) e^{\alpha_t} - \sum_{i:y_i = h_t(x_i)} D_t(i) e^{-\alpha_t}$$

Boosting

So if we minimize Z_t we can minimize bound on training error.

$$Z_t = \sum_{i:y_i \neq h_t(x_i)} D_t(i) e^{\alpha_t} + \sum_{i:y_i = h_t(x_i)} D_t(i) e^{-\alpha_t}$$

$$\frac{\partial Z_t}{\partial \alpha_t} = \sum_{i:y_i \neq h_t(x_i)} D_t(i) e^{\alpha_t} - \sum_{i:y_i = h_t(x_i)} D_t(i) e^{-\alpha_t}$$

$$= e^{\alpha_t} \sum_{i:y_i \neq h_t(x_i)} D_t(i) - e^{-\alpha_t} \sum_{i:y_i = h_t(x_i)} D_t(i)$$

Boosting

So if we minimize Z_t we can minimize bound on training error.

Note that $\sum_{i=1}^{m} D_t(i) = 1$

Boosting

So if we minimize Z_t we can minimize bound on training error.

$$\text{Note that } \sum_{i=1}^{m} D_t(i) = 1$$

$$\text{So: } \sum_{i:y_i \neq h_t(x_i)} D_t(i) + \sum_{i:y_i = h_t(x_i)} D_t(i) = 1$$

Boosting

So if we minimize Z_t we can minimize bound on training error.

Note that $\sum_{i=1}^{m} D_t(i) = 1$

So: $\sum_{i:y_i \neq h_t(x_i)} D_t(i) + \sum_{i:y_i = h_t(x_i)} D_t(i) = 1$

And: $\sum_{i:y_i = h_t(x_i)} D_t(i) = 1 - \sum_{i:y_i \neq h_t(x_i)} D_t(i)$

Boosting

So if we minimize Z_t we can minimize bound on training error.

Note that $\sum_{i=1}^{m} D_t(i) = 1$

So: $\sum_{i:y_i \neq h_t(x_i)} D_t(i) + \sum_{i:y_i = h_t(x_i)} D_t(i) = 1$

And: $\sum_{i:y_i = h_t(x_i)} D_t(i) = 1 - \sum_{i:y_i \neq h_t(x_i)} D_t(i)$

Note that $\sum_{i:y_i \neq h_t(x_i)} D_t(i) = \sum_{i=1}^{m} D_t(i)\delta(y_i \neq h_t(x_i)) = \epsilon_t$

Boosting

Thus, we put together:

1) $$\frac{\partial Z_t}{\partial \alpha_t} = e^{\alpha_t} \sum_{i:y_i \neq h_t(x_i)} D_t(i) - e^{-\alpha_t} \sum_{i:y_i = h_t(x_i)} D_t(i)$$

2) $$\sum_{i:y_i = h_t(x_i)} D_t(i) = 1 - \sum_{i:y_i \neq h_t(x_i)} D_t(i)$$

3) $$\sum_{i:y_i \neq h_t(x_i)} D_t(i) = \epsilon_t$$

To get:

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

Boosting

Thus, we put together:

1) $$\frac{\partial Z_t}{\partial \alpha_t} = e^{\alpha_t} \sum_{i:y_i \neq h_t(x_i)} D_t(i) - e^{-\alpha_t} \sum_{i:y_i = h_t(x_i)} D_t(i)$$

Boosting

Thus, we put together:

1) $$\frac{\partial Z_t}{\partial \alpha_t} = e^{\alpha_t} \sum_{i:y_i \neq h_t(x_i)} D_t(i) - e^{-\alpha_t} \sum_{i:y_i = h_t(x_i)} D_t(i)$$

2) $$\sum_{i:y_i \neq h_t(x_i)} D_t(i) = 1 - \sum_{i:y_i = h_t(x_i)} D_t(i)$$

Boosting

Thus, we put together:

1) $$\frac{\partial Z_t}{\partial \alpha_t} = e^{\alpha_t} \sum_{i:y_i \neq h_t(x_i)} D_t(i) - e^{-\alpha_t} \sum_{i:y_i = h_t(x_i)} D_t(i)$$

2) $$\sum_{i:y_i \neq h_t(x_i)} D_t(i) = 1 - \sum_{i:y_i = h_t(x_i)} D_t(i)$$

3) $$\sum_{i:y_i \neq h_t(x_i)} D_t(i) = \epsilon_t$$

Boosting

Thus, we put together:

1) $$\frac{\partial Z_t}{\partial \alpha_t} = e^{\alpha_t} \sum_{i:y_i \neq h_t(x_i)} D_t(i) - e^{-\alpha_t} \sum_{i:y_i = h_t(x_i)} D_t(i)$$

2) $$\sum_{i:y_i \neq h_t(x_i)} D_t(i) = 1 - \sum_{i:y_i = h_t(x_i)} D_t(i)$$

3) $$\sum_{i:y_i \neq h_t(x_i)} D_t(i) = \epsilon_t$$

To get:

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

Boosting

So if we minimize Z_t we can minimize bound on training error.

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

Boosting

So if we minimize Z_t we can minimize bound on training error.

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

Boosting

So if we minimize Z_t we can minimize bound on training error.

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

$$0 = e^{\alpha_t}\epsilon_t - e^{-\alpha_t}(1-\epsilon_t)$$

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

$$(1-\epsilon_t) = e^{2\alpha_t}\epsilon_t$$

Boosting

So if we minimize Z_t we can minimize bound on training error.

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

Boosting

So if we minimize Z_t we can minimize bound on training error.

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

$$0 = e^{\alpha_t}\epsilon_t - e^{-\alpha_t}(1-\epsilon_t)$$

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

$$(1-\epsilon_t) = e^{2\alpha_t}\epsilon_t$$

$$\frac{1-\epsilon_t}{\epsilon_t} = e^{2\alpha_t}$$

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

Boosting

So let's finish our proof on the bound of training error.

We've established:

1) $$\frac{1}{m}\sum_{i=1}^{m}\delta(H(x_i)\neq y_i)\leq\frac{1}{m}\sum_{i=1}^{m}e^{-y_i f(x_i)}=\prod_{t=1}^{T}Z_t$$

Boosting

So let's finish our proof on the bound of training error.

We've established:

1) $$\frac{1}{m}\sum_{i=1}^{m}\delta(H(x_i)\neq y_i)\leq\frac{1}{m}\sum_{i=1}^{m}e^{-y_i f(x_i)}=\prod_{t=1}^{T}Z_t$$

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

Boosting

So let's finish our proof on the bound of training error.

We've established:

1) $$\frac{1}{m}\sum_{i=1}^{m}\delta(H(x_i)\neq y_i)\leq\frac{1}{m}\sum_{i=1}^{m}e^{-y_i f(x_i)}=\prod_{t=1}^{T}Z_t$$

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

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

Boosting

So let's finish our proof on the bound of training error.

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

Boosting

So let's finish our proof on the bound of training error.

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

$$Z_t^2 = \epsilon_t^2 e^{2\alpha_t} + 2\epsilon_t(1 - \epsilon_t) + (1 - \epsilon_t)^2 e^{-2\alpha_t}$$

Boosting

So let's finish our proof on the bound of training error.

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

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

$$\begin{aligned} Z_t^2 &= \epsilon_t^2 e^{2\alpha_t} + 2\epsilon_t(1 - \epsilon_t) + (1 - \epsilon_t)^2 e^{-2\alpha_t} \\ &= \epsilon_t^2 \left(\frac{1 - \epsilon_t}{\epsilon_t} \right) + 2\epsilon_t(1 - \epsilon_t) + (1 - \epsilon_t)^2 \left(\frac{\epsilon_t}{1 - \epsilon_t} \right) \end{aligned}$$

Boosting

So let's finish our proof on the bound of training error.

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

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

$$\begin{aligned} Z_t^2 &= \epsilon_t^2 e^{2\alpha_t} + 2\epsilon_t(1 - \epsilon_t) + (1 - \epsilon_t)^2 e^{-2\alpha_t} \\ &= \epsilon_t^2 \left(\frac{1 - \epsilon_t}{\epsilon_t} \right) + 2\epsilon_t(1 - \epsilon_t) + (1 - \epsilon_t)^2 \left(\frac{\epsilon_t}{1 - \epsilon_t} \right) \\ &= 4\epsilon_t(1 - \epsilon_t) \end{aligned}$$

Boosting

So let's finish our proof on the bound of training error.

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

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

$$\begin{aligned} Z_t^2 &= \epsilon_t^2 e^{2\alpha_t} + 2\epsilon_t(1 - \epsilon_t) + (1 - \epsilon_t)^2 e^{-2\alpha_t} \\ &= \epsilon_t^2 \left(\frac{1 - \epsilon_t}{\epsilon_t} \right) + 2\epsilon_t(1 - \epsilon_t) + (1 - \epsilon_t)^2 \left(\frac{\epsilon_t}{1 - \epsilon_t} \right) \\ &= 4\epsilon_t(1 - \epsilon_t) \\ Z_t &= 2\sqrt{\epsilon_t(1 - \epsilon_t)} = \sqrt{1 - (1 - 2\epsilon_t)^2} \end{aligned}$$

Boosting

So let's finish our proof on the bound of training error.

Training error of final classifier is bounded by:

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

Using $1-x \leq e^{-x}$

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

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) !!

From class slides.