COGS 118A, Winter 2020

Supervised Machine Learning Algorithms

Lecture 10: Support Vector Machine

Zhuowen Tu

Support Vector Machine

Linear Model

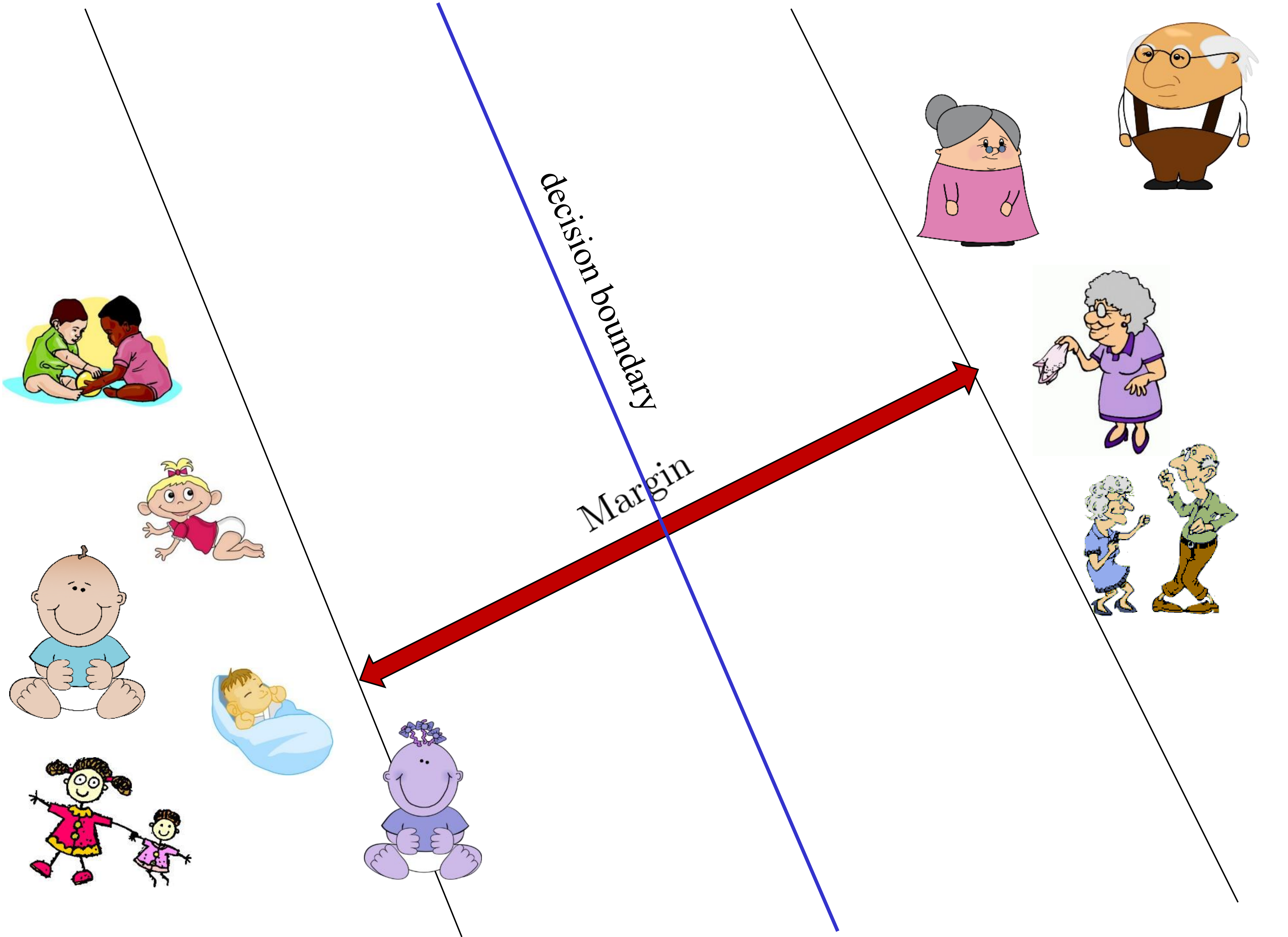
A linear model:

$$f(\mathbf{x}; \mathbf{w}, b) = < \mathbf{w}, \mathbf{x} > + b$$
$$= \mathbf{w} \cdot \mathbf{x} + b$$
$$= \mathbf{w}^T \mathbf{x} + b$$

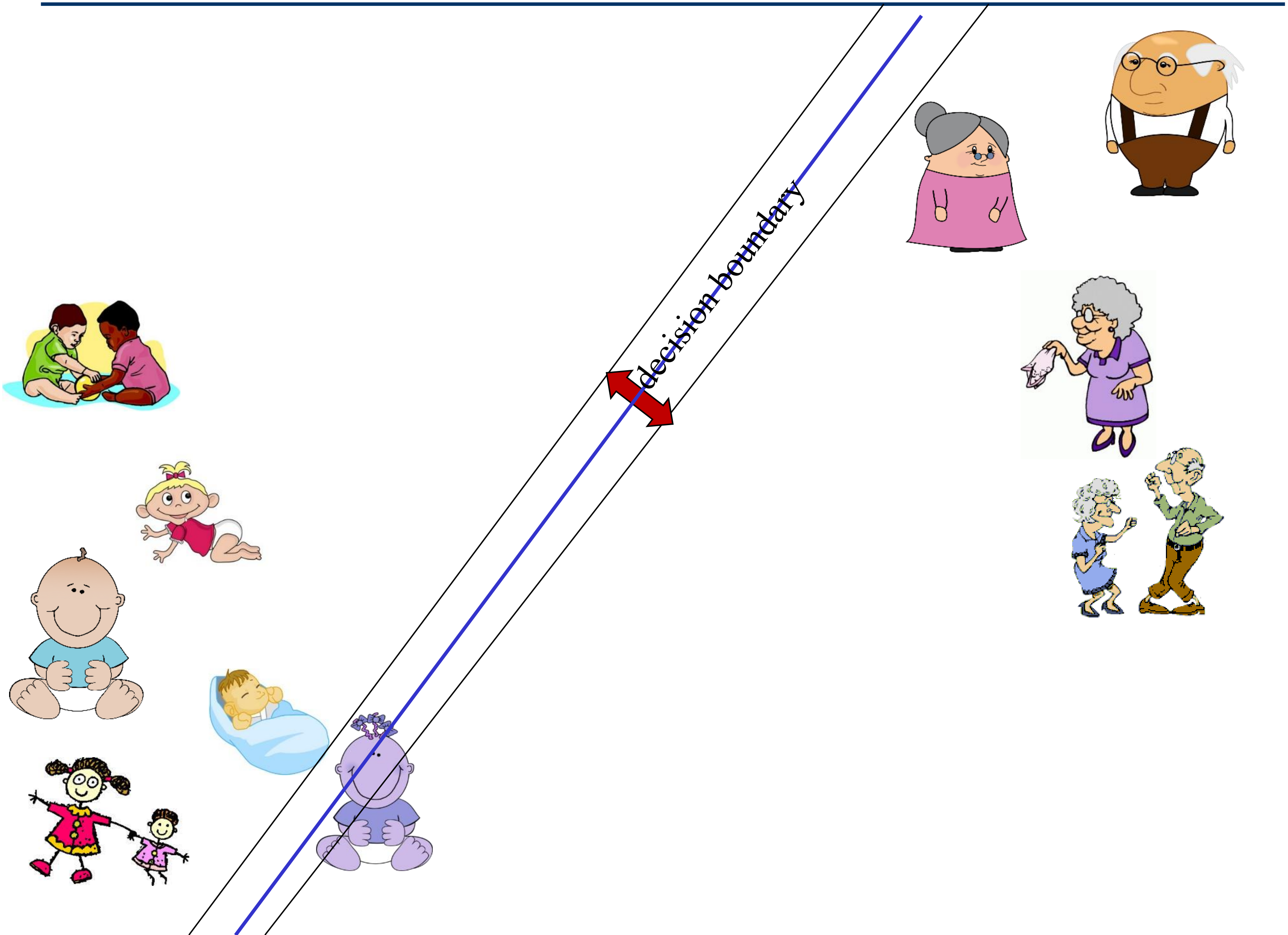
$\mathbf{x} = \mathbb{R}^m$ $\mathbf{w} = \mathbb{R}^m$ $b \in \mathbb{R}$

This is a linear function and our job is find the optimal **w** and b to best fit the prediction in learning.

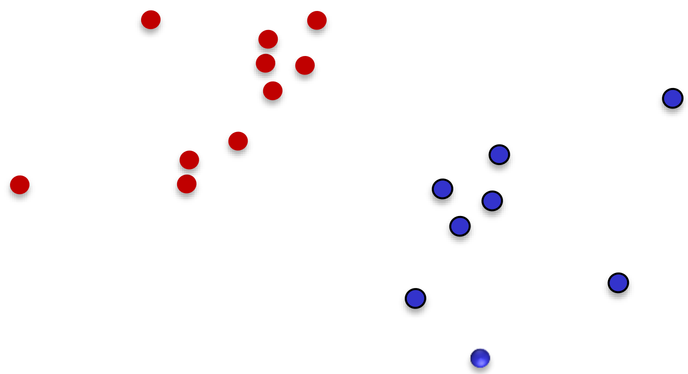
Why large margin?



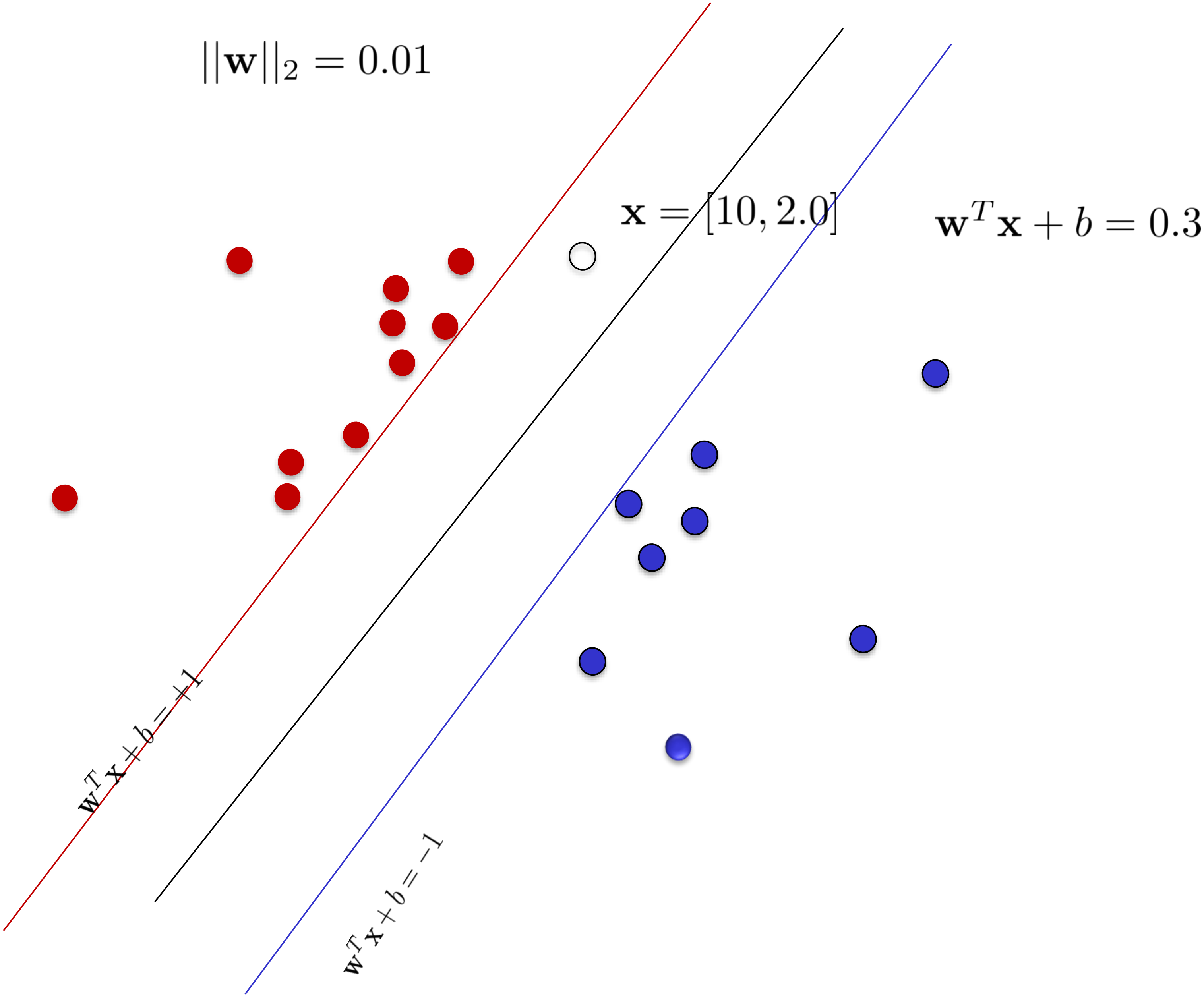
Why large margin?



How to understand

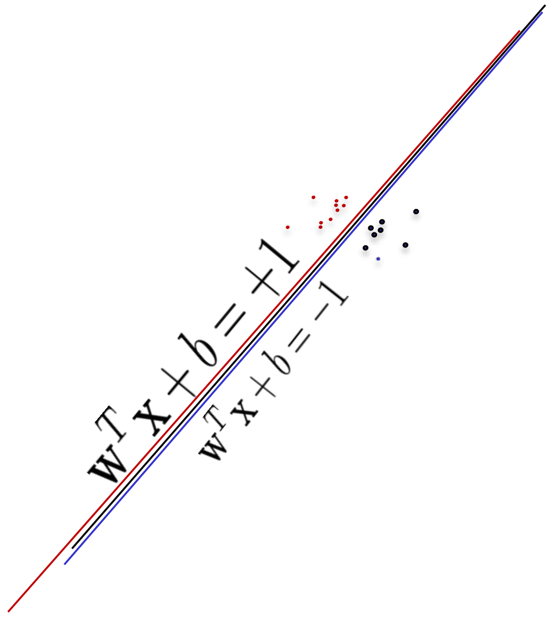


How to understand: a large margin



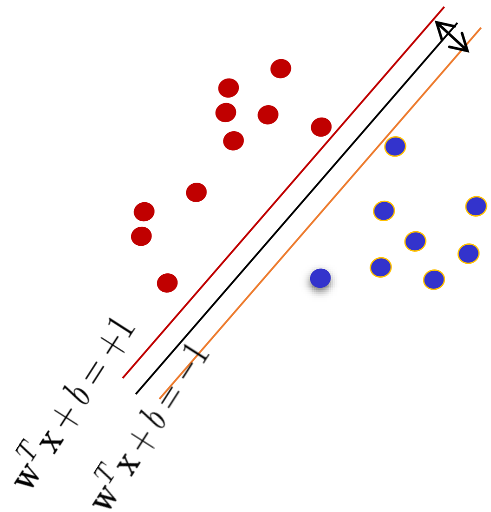
How to understand: a small margin

$$||\mathbf{w}||_2 = 10,000$$



Why margin?

$$e_{testing} \leq e_{training} + \sqrt{\frac{h(\log(2n/h+1))-\log(\eta/4)}{n}}$$



$$M = \frac{2}{\sqrt{\mathbf{w}^T \mathbf{w}}}$$

$$\mathbf{w}^T \mathbf{w} = ||\mathbf{w}||^2$$

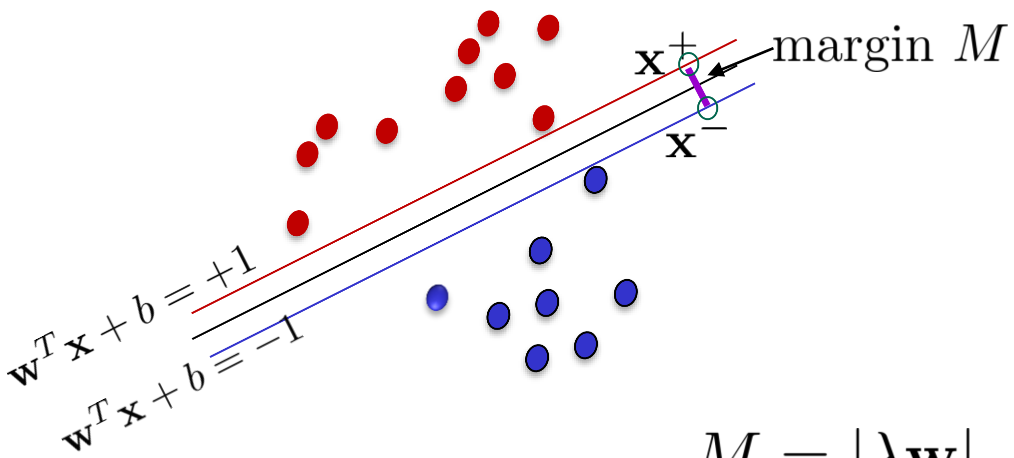
Find: $\arg\min_{\mathbf{w}} C \times (\#training\ errors) + \frac{1}{2}||\mathbf{w}||^2$

Why is $\sqrt{\frac{h(\log(2n/h+1))-\log(\eta/4)}{n}}$ related to $||\mathbf{w}||^2$?

In machine learning, a term called "regularization", has been frequently used to prevent overfitting.

"Margin" is a term researchers typically use to "regularize" the underlying classifier (there are of course other ways to impose regularization https://en.wikipedia.org/wiki/Regularization_(mathematics)).

Computing the margin width



$$M = |\lambda \mathbf{w}|$$

$$\mathbf{w}^T\mathbf{x}^- + b = -1$$

$$\lambda = \frac{2}{\mathbf{w}^T\mathbf{w}} \qquad \lambda \in \mathbb{R}$$

$$\mathbf{x}^+ = \mathbf{x}^- + \lambda \mathbf{w}$$

$$||\mathbf{w}||_2 = \sqrt{\mathbf{w}^T\mathbf{w}}$$

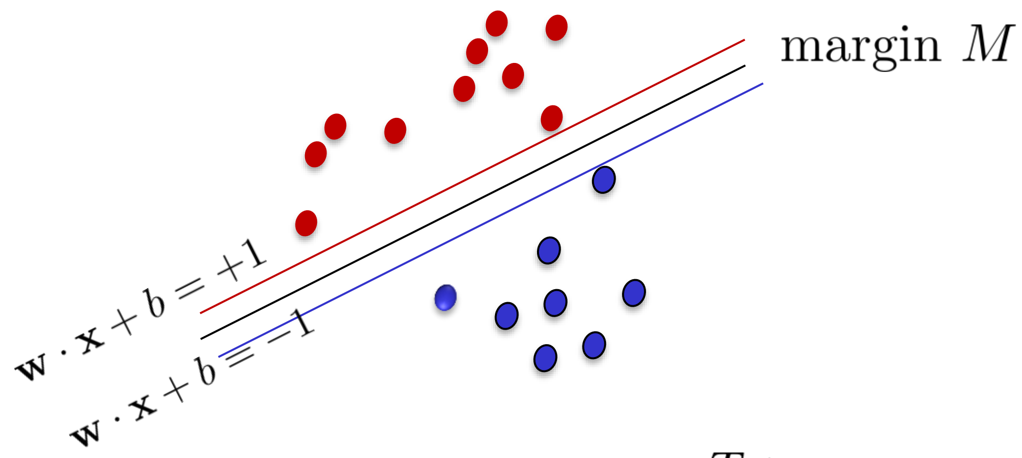
$$\mathbf{w}^T\mathbf{x}^+ + b = +1$$

$$\Downarrow$$

$$\text{Margin: } M = ||\mathbf{x}^+ - \mathbf{x}^-||_2$$
$$= ||\lambda \mathbf{w}||_2 \in \mathbb{R}$$

$$M = ||\lambda \mathbf{w}||_2 = \frac{2\sqrt{\mathbf{w}^T\mathbf{w}}}{\mathbf{w}^T\mathbf{w}}$$
$$= \frac{2}{\sqrt{\mathbf{w}^T\mathbf{w}}}$$

Computing the margin width



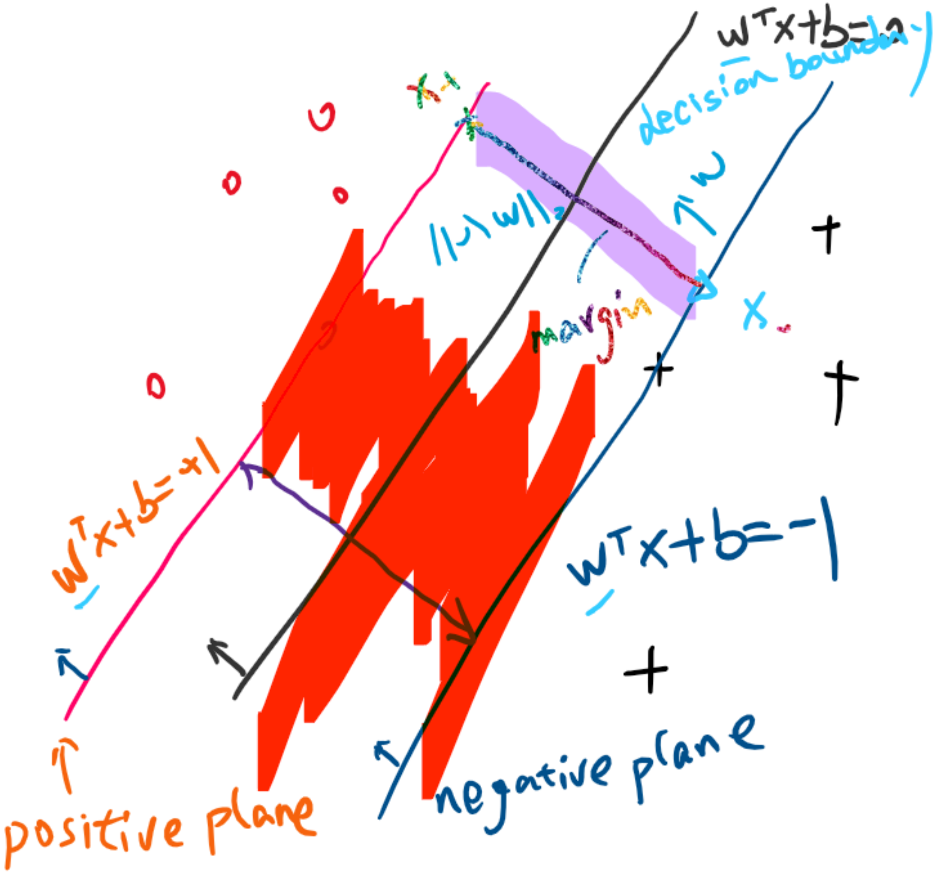
$$\mathbf{w}^T\mathbf{x}^- + b = -1$$

$$\mathbf{x}^+ = \mathbf{x}^- + \lambda\mathbf{w}$$

$$\mathbf{w}^T\mathbf{x}^+ + b = +1$$

$$\text{Margin: } M = ||\mathbf{x}^+ - \mathbf{x}^-||_2$$
$$= ||\lambda\mathbf{w}||_2 \in \mathbb{R}$$

$$\mathbf{w}^T(\mathbf{x}^- + \lambda\mathbf{w}) + b = +1$$
$$\Downarrow$$
$$\mathbf{w}^T\mathbf{x}^- + \mathbf{w}^T\lambda\mathbf{w} + b = +1$$
$$+$$
$$\mathbf{w}^T\mathbf{x}^- + b = -1$$
$$\Downarrow$$
$$\lambda\mathbf{w}^T\mathbf{w} = 2 \qquad \lambda \in \mathbb{R}$$
$$\Downarrow$$
$$\lambda = \frac{2}{<\mathbf{w},\mathbf{w}>}$$



classifier: $\boxed{\text{sign}(w^Tx+b)}$

separable

concept

margin → no-margin (and

⇓

? margin

✓ ✓

$\boxed{w^Tx_- + b = -1}$

$x_+ = x_- + \lambda w$

? ? ? ✓

not unit vector

$\lambda \in \mathbb{R}$

$$\begin{cases} w^T x_- + b = -1 \\ w^T x_+ + b = +1 \end{cases}$$

$$x_+ = x_- + \lambda w$$

$$w^T(x_- + \lambda w) + b = +1$$

$$w^T x_- + b + w^T \lambda w = +1$$

$$\underbrace{w^T x_- + b}_{-1} + \lambda w^T w = +1$$

$$\lambda w^T w = +1 + 1$$

w is given

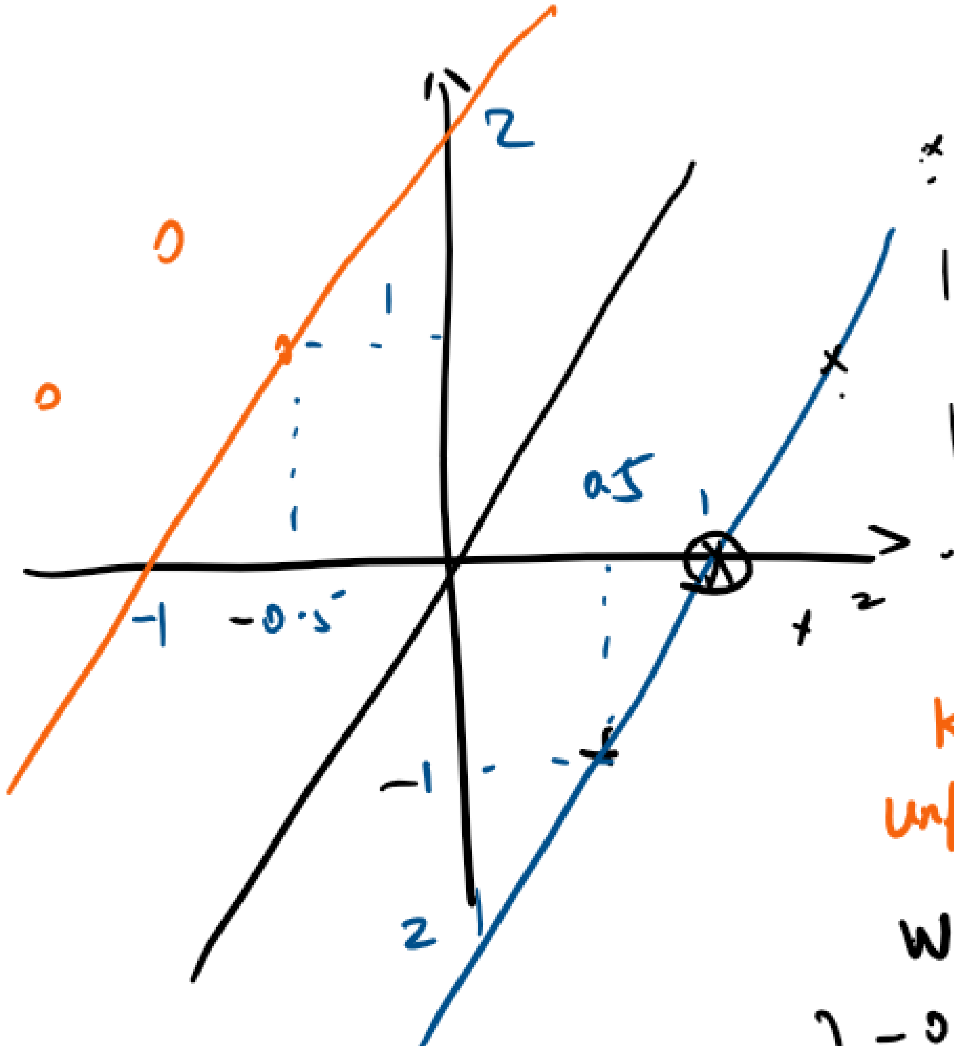
$$\lambda w^T w = 2$$

$$\|w\|_2 = \sqrt{w^T w} \qquad \|w\|_2^2 = w^T w \in \mathbb{R}$$

$$\lambda = \frac{2}{\|w\|_2^2}$$

margin: $\|\lambda w\|_2 = \lambda \|w\|_2 = \frac{2\|w\|_2}{\|w\|_2^2} = \frac{2\sqrt{w^T w}}{w^T w}$

$$= \frac{2}{\sqrt{w^T w}} \quad \square$$



$w^T x + b = +1$

$w^T = (w_1, w_2)$

$w = \begin{pmatrix} w_1 \\ w_2 \end{pmatrix} \quad x = \begin{pmatrix} x_1 \\ x_2 \end{pmatrix}$

$w_1 x_1 + w_2 x_2 + b = +1$

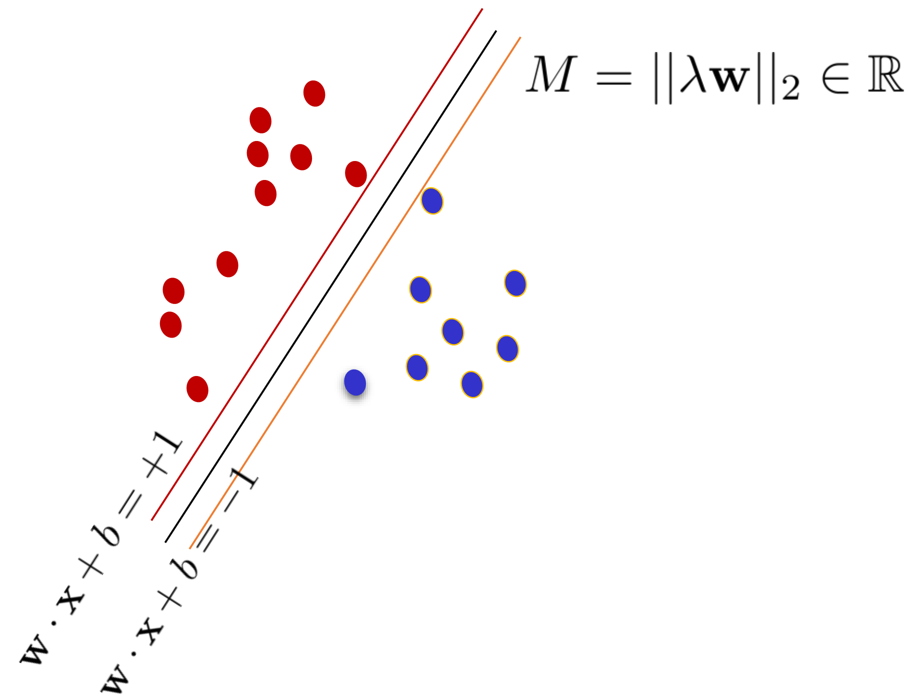
known: x_1, x_2

unknown: w_1, w_2, b

$w_1 \times 0.5 + w_2 \times 1 + b = +1$

$$\begin{cases} -0.5 w_1 + w_2 + b = +1 \\ w_1 \times 0.5 - w_2 + b = -1 \\ 2 w_1 + 4 w_2 + b = -1 \end{cases}$$

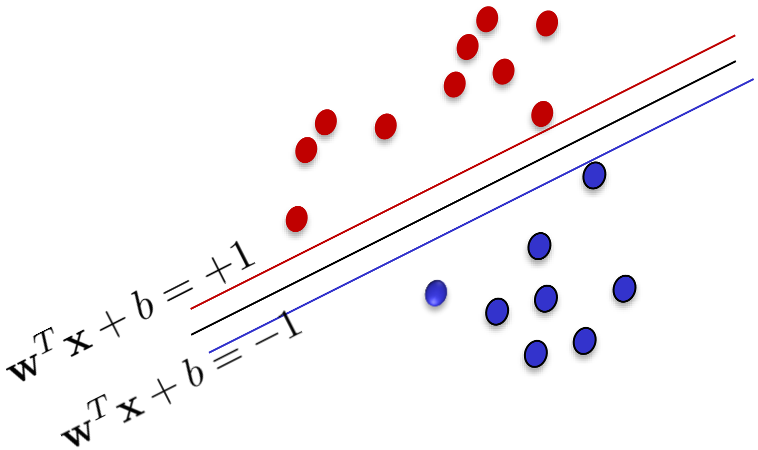
What would be the maximum margin



What would **w** look like if we just want to increase the margin?

A. Unit vector
B. Infinitely small magnitude
C. Infinitely large magnitude

Training SVM using gradient descent



margin $M = \frac{2}{\sqrt{\mathbf{w}^T\mathbf{w}}}$

Separable case: all positive and negative points are perfectly separable.

Maximizing $\frac{2}{\sqrt{\mathbf{w}^T\mathbf{w}}}$ is equivlant to minimizing $\mathbf{w}^T\mathbf{w} = ||\mathbf{w}||^2$

$$S_{training} = \{(\mathbf{x}_i, y_i), i = 1..n\}$$

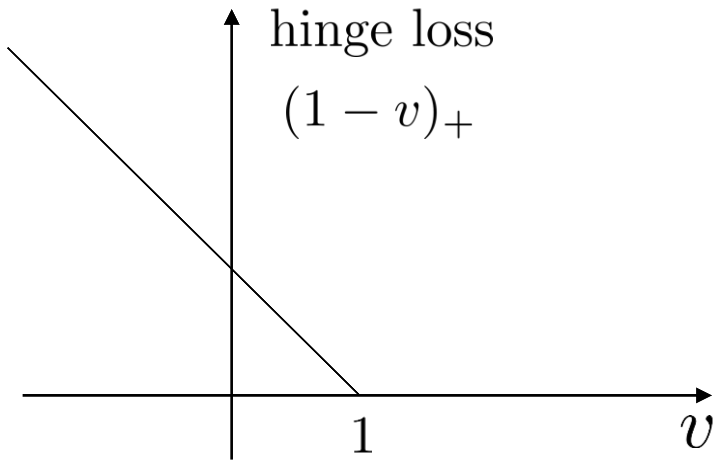
Find: $\arg\min_{\mathbf{w}} \frac{1}{2}||\mathbf{w}||^2$ subject to $y_i(\mathbf{w}^T\mathbf{x} + b) - 1 \geq 0$

Hinge Loss

Find: $\arg\min_{\mathbf{w}} \frac{1}{2}||\mathbf{w}||^2$

subject to $y_i(\mathbf{w}^T\mathbf{x} + b) - 1 \geq 0$

Hinge: $(1 - v)_+ = max(0, 1 - v)$

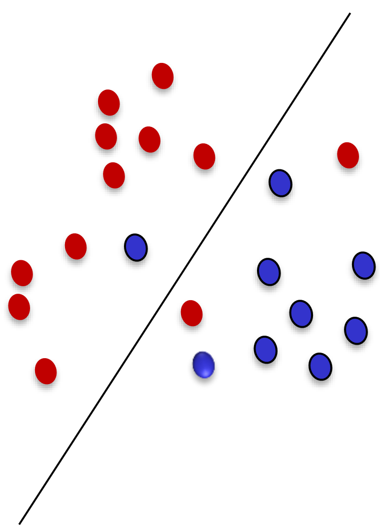


Find: $\arg\min_{\mathbf{w}} \frac{1}{2}||\mathbf{w}||^2 + C \times \sum_{i=1}^{n}(1 - y_i \times (\mathbf{w}^T\mathbf{x}_i + b))_+$

SVM: non-separable

Now let's consider non-separable case:

- $denotes +1$ (red)
- $denotes -1$ (blue)



Find minimum $\mathbf{w} \cdot \mathbf{w}$,
while minimizing the number of miss-classified samples.

Problem: minimizing two things
makes the task problematic.

$$e_{testing} \leq e_{training} + bound(generalization(f))$$

training error

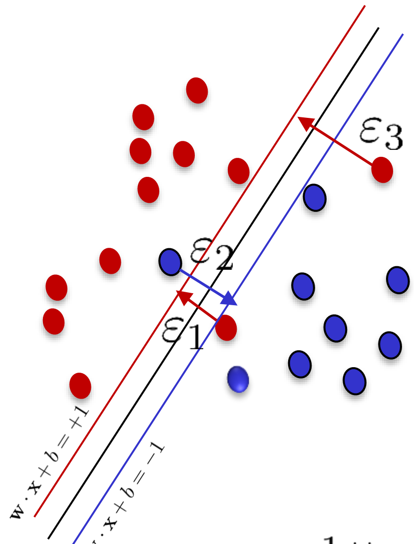
generalization error

$$\text{Find: } \arg\min_{\mathbf{w}} C \times (\#training\ errors) + \frac{1}{2}||\mathbf{w}||^2$$

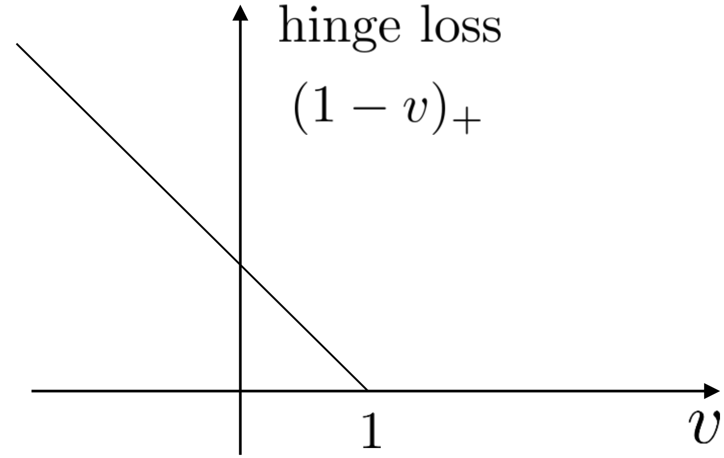
tradeoff parameter

Doable but not ideal!

SVM



$$M = \frac{2}{||\mathbf{w}||_2}$$



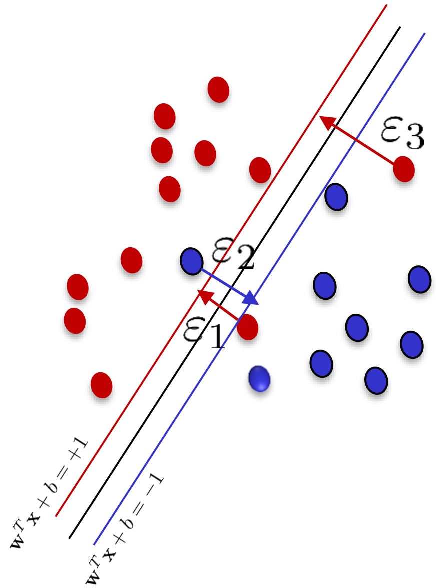
Find: $\arg\min_{\mathbf{w}} \frac{1}{2}||\mathbf{w}||^2 + C \times \sum_{i=1}^{n} \varepsilon_i \qquad \varepsilon_i \geq 0, \forall i$

subject to: $y_i \times (\mathbf{w}^T \mathbf{x}_i + b) \geq 1 - \varepsilon_i$

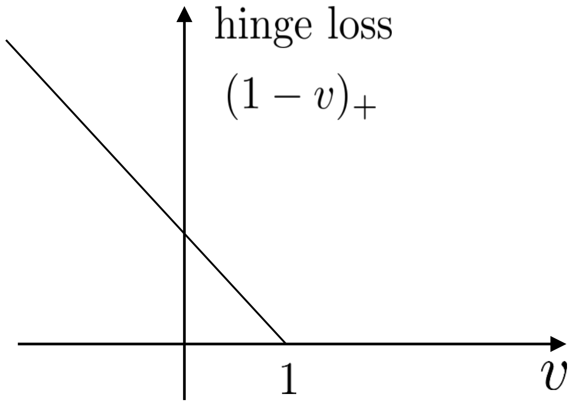
Find: $\arg\min_{\mathbf{w},b} \frac{1}{2}||\mathbf{w}||^2 + C \times \sum_{i=1}^{n} max(0, 1 - y_i(\mathbf{w}^T \mathbf{x}_i + b))$

Find: $\arg\min_{\mathbf{w},b} \frac{1}{2}||\mathbf{w}||^2 + C \times \sum_{i=1}^{n} (1 - y_i(\mathbf{w}^T \mathbf{x}_i + b))_+$

SVM: non-separable



$$M = \frac{2}{||\mathbf{w}||_2}$$

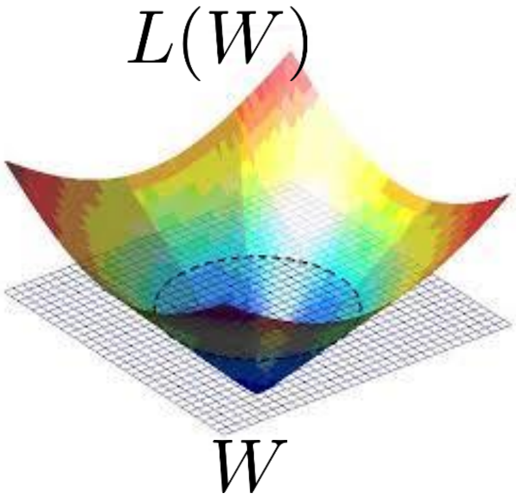


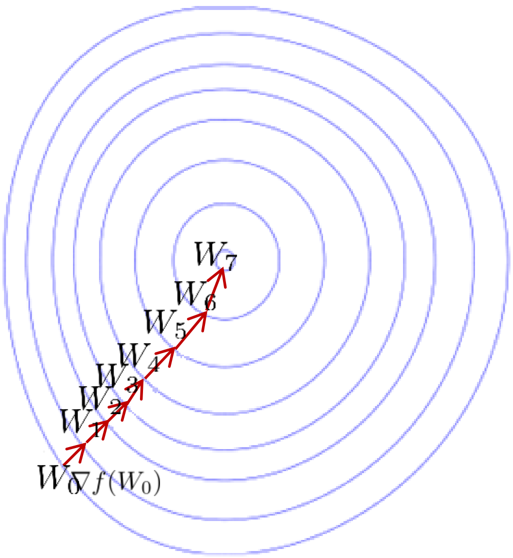
$$\text{Minimize } \mathcal{L}(\mathbf{w}, b) = \frac{1}{2}||\mathbf{w}||^2 + C\sum_{i=1}^{n}(1 - y_i(\mathbf{w}^T\mathbf{x}_i + b))_+$$

$$\frac{\mathcal{L}(\mathbf{w},b)}{\partial \mathbf{w}} = \mathbf{w} + C\sum_{i=1}^{n}\begin{cases} 0 & if \;\; y_i(\mathbf{w}^T\mathbf{x}_i + b) \geq 1 \\ -y_i\mathbf{x}_i & otherwise \end{cases}$$

$$\frac{\mathcal{L}(\mathbf{w},b)}{\partial b} = C\sum_{i=1}^{n}\begin{cases} 0 & if \;\; y_i(\mathbf{w}^T\mathbf{x}_i + b) \geq 1 \\ -y_i & otherwise \end{cases}$$

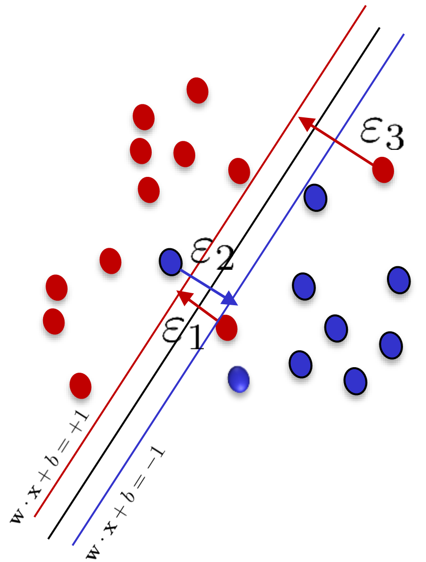
Gradient descent



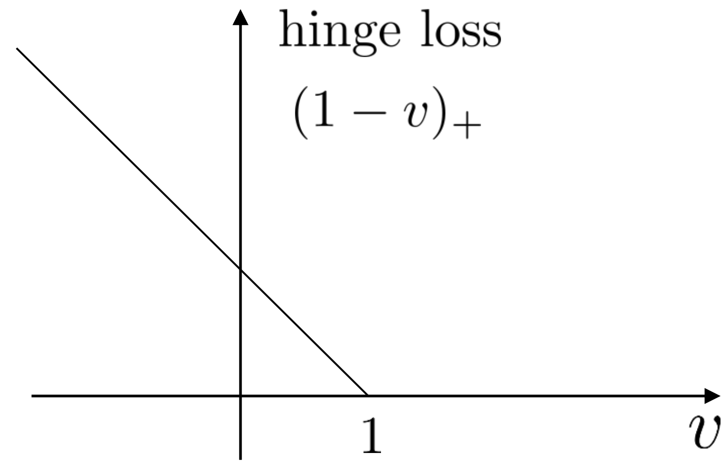


$$W_{t+1} \leftarrow W_t - \lambda_t \nabla L(W_t) \quad \lambda_t : stepsize$$

Convex?



$$M = \frac{2}{||\mathbf{w}||_2}$$



$$\text{Find: } \arg\min_{\mathbf{w},b} \frac{1}{2}||\mathbf{w}||^2 + C\sum_{i=1}^{n}(1-y_i(\mathbf{w}^T\mathbf{x}_i+b))_+$$

A. Convex
B. Concave
C. No-convex
D. It depends

The summation of convex functions is also convex.

What's the difference between logistic regression and linear SVM?

Logistic regression:

SVM:

A. They are different in both training and testing.

B. They differ in training but are the same in testing.

C. They differ in testing but are the same in training.

D. They are completely different.

What's the difference between logistic regression and linear SVM?

Logistic regression:

$$p(y|\mathbf{x}) = \frac{1}{1+e^{-y(\mathbf{w}^T\mathbf{x}+b)}}$$

Training: $\arg\min_{(\mathbf{w},b)} \sum_{i=1}^{n} \ln(1 + e^{-y_i(\mathbf{w}^T\mathbf{x}_i+b)})$

Test: $f(\mathbf{x};\mathbf{w},b) = \begin{cases} +1 & if\ \frac{1}{1+e^{-(\mathbf{w}^T\mathbf{x}+b)}} \geq 0.5 \\ -1 & otherwise \end{cases}$

Equivalent to: $f(\mathbf{x};\mathbf{w},b) = \begin{cases} +1 & if\ \mathbf{w}^T\mathbf{x}+b \geq 0 \\ -1 & otherwise \end{cases}$

SVM:

Training: $\arg\min_{\mathbf{w},b} \frac{1}{2}||\mathbf{w}||^2 + C\sum_{i=1}^{n}(1 - y_i(\mathbf{w}^T\mathbf{x}_i + b))_+$

Test: $f(\mathbf{x};\mathbf{w},b) = \begin{cases} +1 & if\ \mathbf{w}^T\mathbf{x}+b \geq 0 \\ -1 & otherwise \end{cases}$

A. They are different in both training and testing.

B. They differ in training but are the same in testing.

C. They differ in testing but are the same in training.

D. They are completely different.

Main motivation

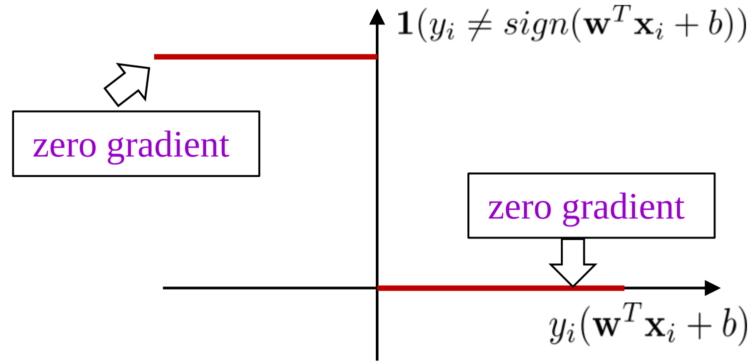
Hard->Half-hard->Soft

Error

Standard loss (error) function

Standard 0/1 loss (gradient 0 nearly everywhere, no gradient feedback):

Training: Minimize $\mathcal{L}(\mathbf{w}, b) = \sum_i \mathbf{1}(y_i \neq sign(\mathbf{w}^T\mathbf{x}_i + b))$



It is the most directly loss, but is also the hardest to minimize.

Zero gradient everywhere!

Main motivation

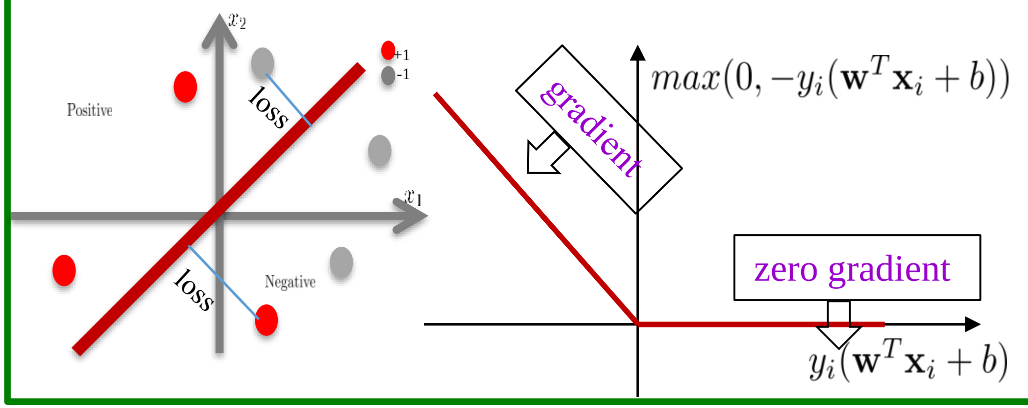
Hard->Half-hard->Soft

Error

Half-hard loss (error) function

Loss implicitly used in the perceptron algorithm: with gradient feedback when the target (ground-truth label) and the output (classification) are different).

Training: Minimize $\mathcal{L}(\mathbf{w}, b) = \sum_i max(0, -y_i(\mathbf{w}^T\mathbf{x}_i + b))$



Zero loss for correct classification (no gradient).

A loss based on the distance to the decision boundary for misclassification (with gradient).

Used in the perceptron training.

Main motivation

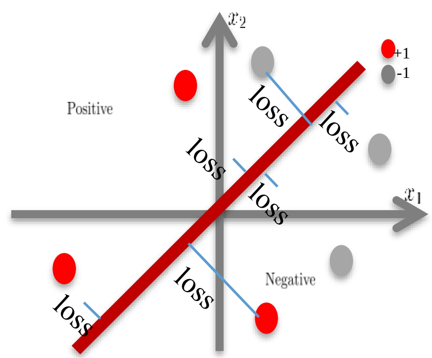
Hard->Half-hard->Soft

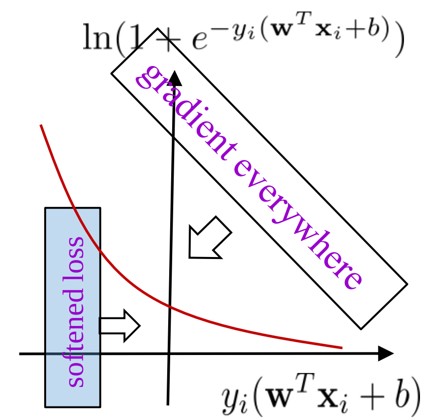
Error

Soft loss (error) function

Loss used in logistic regression.

Training: minimize $\mathcal{L}(\mathbf{w}, b) = \sum_{i=1}^{n} \ln(1 + e^{-y_i(\mathbf{w}^T\mathbf{x}_i + b)})$





Every data point receives a loss (gradient everywhere).

A loss based on the distance to the decision boundary for wrong classification (has a gradient).

Used in logistic regression classifier.

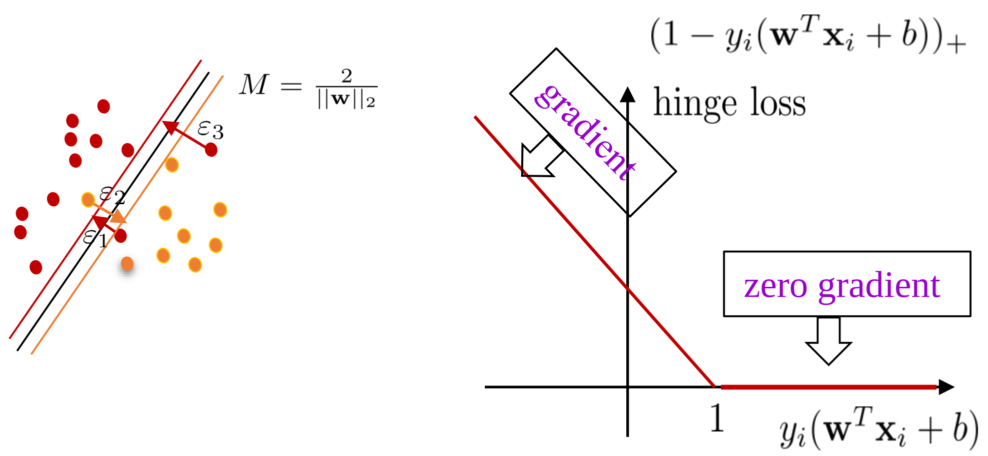
Main motivation

Hard->Hinge

Error

Loss in SVM

Minimize $\mathcal{L}(\mathbf{w}, b) = \frac{1}{2}||\mathbf{w}||^2 + C\sum_{i=1}^{n}(1 - y_i(\mathbf{w}^T\mathbf{x}_i + b))_+$



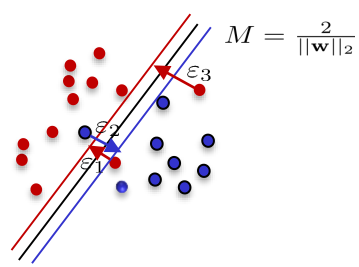
Zero loss for correct classification beyond the margin (no gradient).

A loss based on the distance to the decision boundary for misclassification or within the margin (with gradient).

Recap: Support Vector Machine

Intuition: It explicitly introduces a "regularization" (margin) into the objective function to combine with a classification error (restricted using a hinge loss) term.

- It achieves unprecedented robustness when training a linear classifier due to the use of margin term in training.
- The learned model is based on a balance between classification error and margin. The balancing term C is typically attained using cross-validation.
- Kernel based SVM makes non-separable samples feasible to classify by projecting the data onto higher dimensional spaces.
- The features defined under kernels don't need to be computed explicitly.
- The learned weights $\mathbf{w}$ is carried in the weights for the samples and those samples with non-zero weights are called support vectors.



Recap: Support Vector Machine

Math:

Training :

$$\text{Minimize } \mathcal{L}(\mathbf{w}, b) = \frac{1}{2}||\mathbf{w}||^2 + C\sum_{i=1}^{n}(1 - y_i(\mathbf{w}^T\mathbf{x}_i + b))_+$$

$$\frac{\mathcal{L}(\mathbf{w},b)}{\partial \mathbf{w}} = \mathbf{w} + C\sum_{i=1}^{n}\begin{cases}0 & if \;\; y_i(\mathbf{w}^T\mathbf{x}_i + b) \geq 1 \\ -y_i\mathbf{x}_i & otherwise\end{cases}$$

$$\frac{\mathcal{L}(\mathbf{w},b)}{\partial b} = C\sum_{i=1}^{n}\begin{cases}0 & if \;\; y_i(\mathbf{w}^T\mathbf{x}_i + b) \geq 1 \\ -y_i & otherwise\end{cases}$$

Testing :

$$f(\mathbf{x}; \mathbf{w}, b) = \begin{cases}+1 & if \; \mathbf{w}^T\mathbf{x} + b \geq 0 \\ -1 & otherwise\end{cases}$$

Implementation:

Gradient Descent Direction

(a) Pick a direction $\nabla\mathcal{L}(\mathbf{w}_t, b_t)$

(b) Pick a step size λ_t

(c) $\mathbf{w}_{t+1} = \mathbf{w}_t - \lambda_t \times \nabla\mathcal{L}_{\mathbf{w}_t}(\mathbf{w}_t, b_t)$ such that function decreases;
$b_{t+1} = b_t - \lambda_t \times \nabla\mathcal{L}_{b_t}(\mathbf{w}_t, b_t)$

(d) Repeat