Support Vector Machines (SVMs)

Aarti Singh

Machine Learning 10-315
Oct 21, 2020

Discriminative Classifiers

Optimal Classifier:

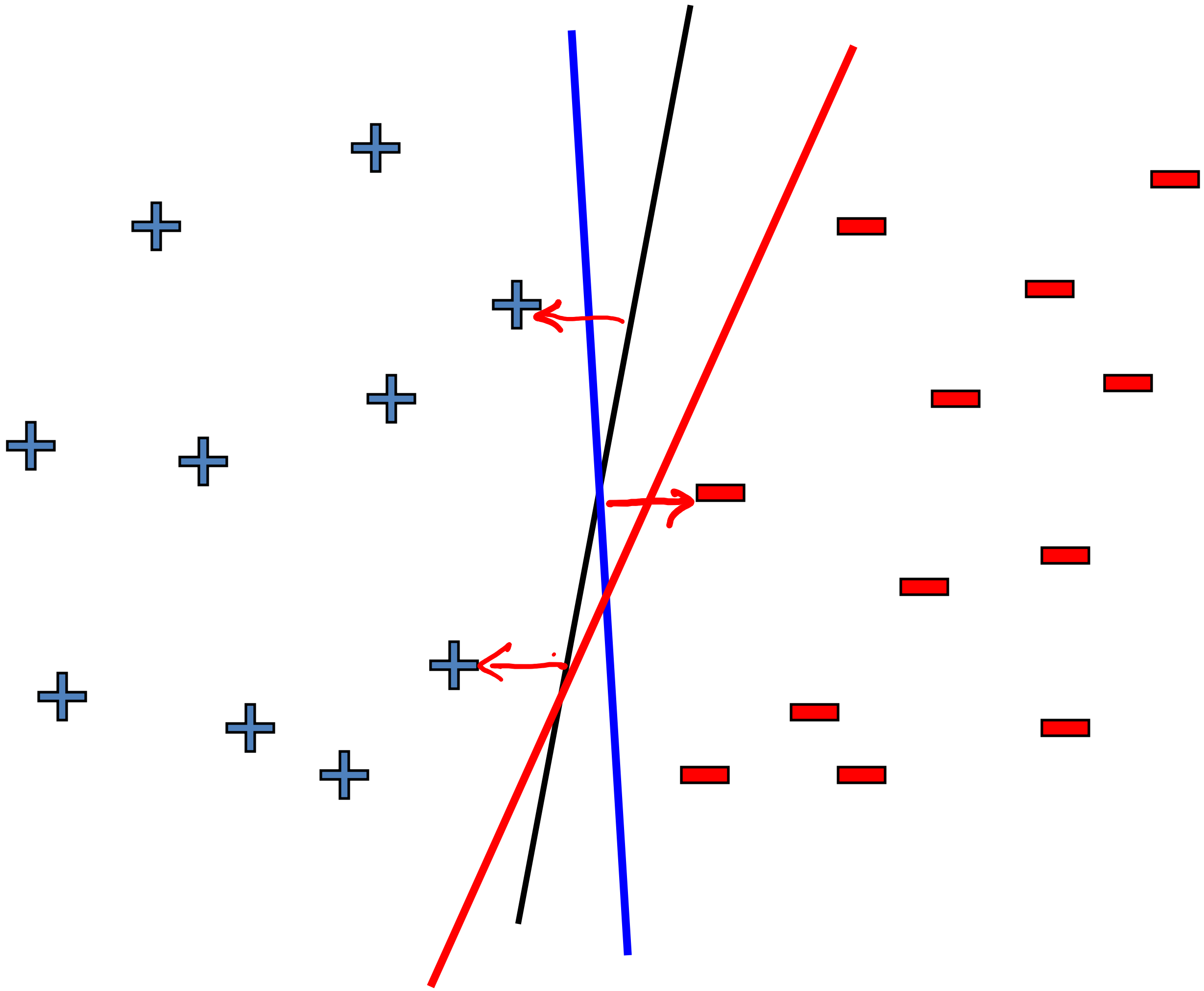
$$f^*(x) = \arg\max_{Y=y} P(Y = y|X = x)$$
$$= \arg\max_{Y=y} P(X = x|Y = y)P(Y = y)$$

Why not learn P(Y|X) directly? Or better yet, why not learn the decision boundary directly?

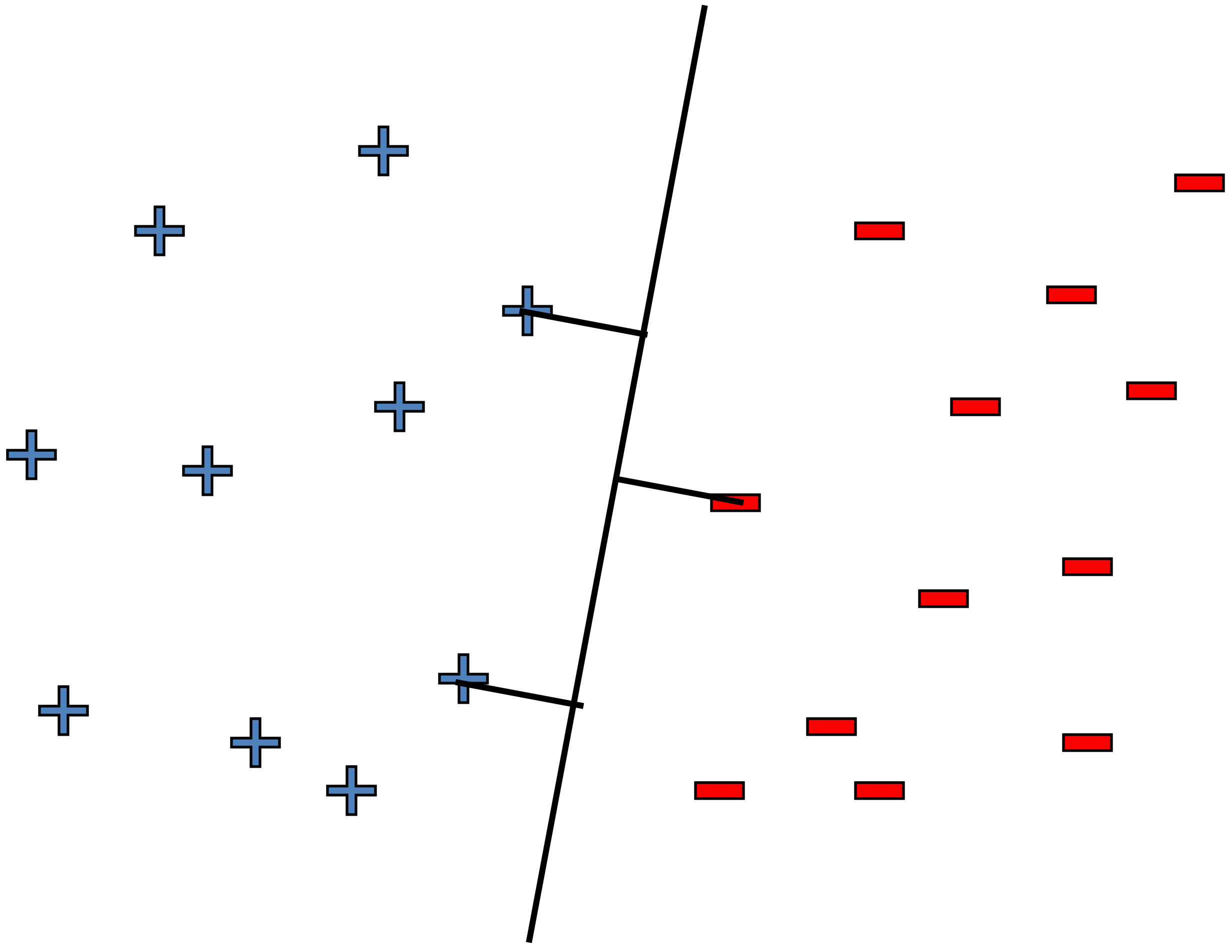
- Assume some functional form for P(Y|X) (e.g. Logistic Regression) or for the decision boundary (e.g. Neural nets, SVMs)
- Estimate parameters of functional form directly from training data

At Pittsburgh G-20 summit …

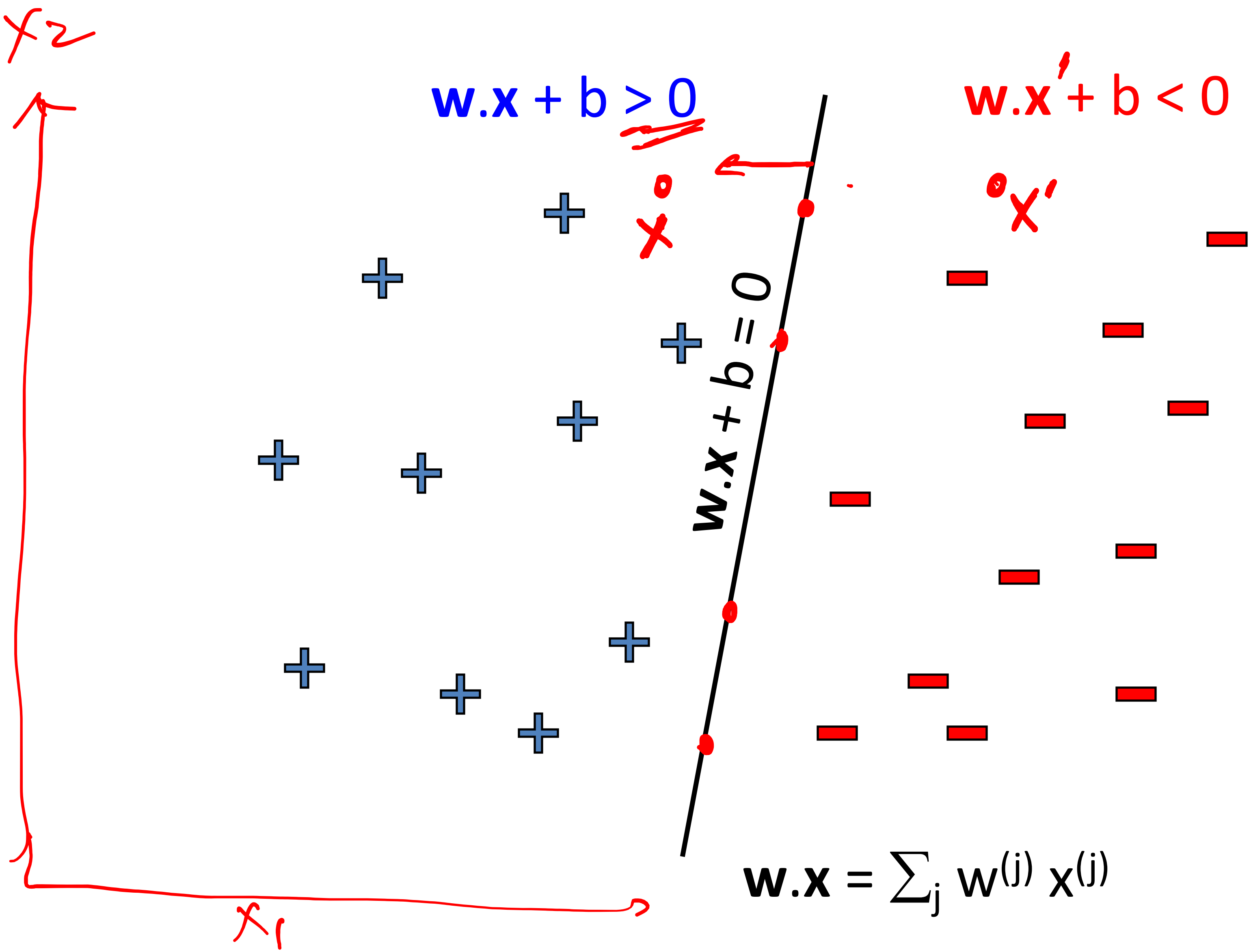
Linear classifiers – which line is better?



Pick the one with the largest margin!

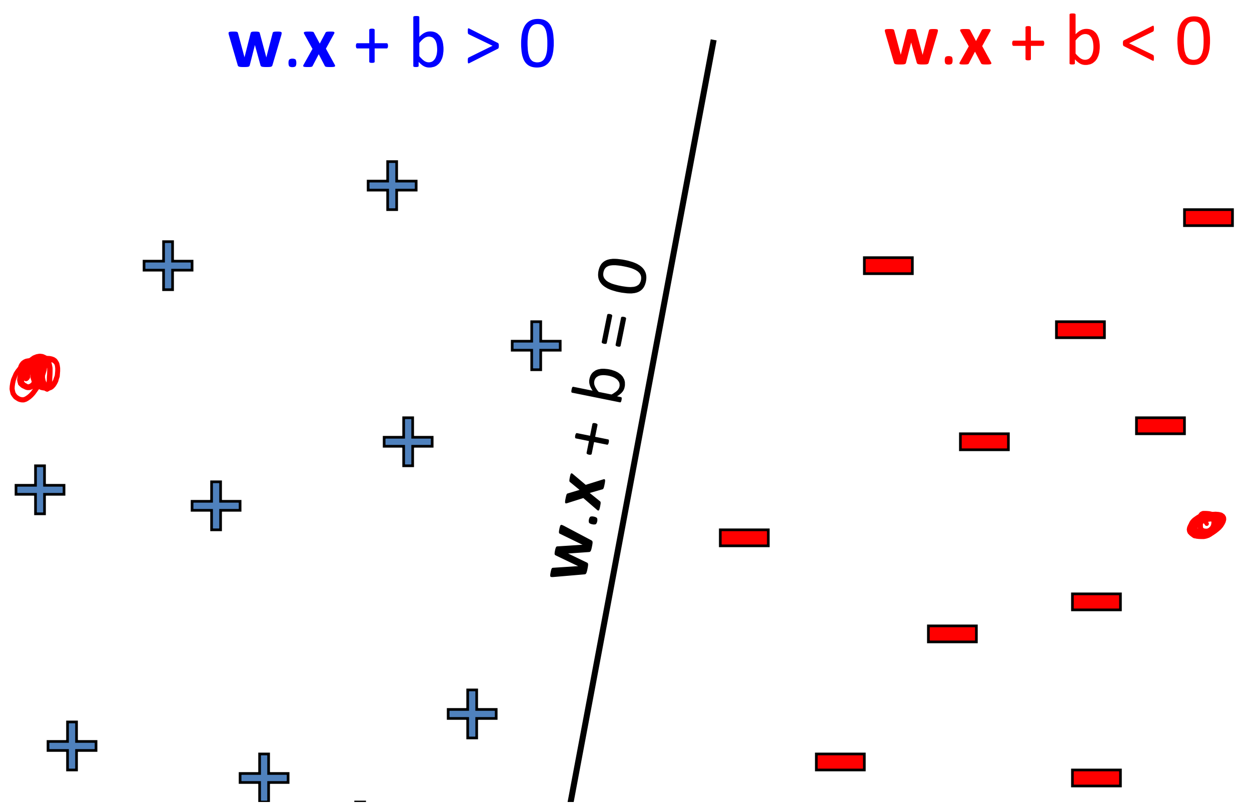


Parameterizing the decision boundary



$\mathbf{w}.\mathbf{x} = \sum_j w^{(j)} x^{(j)}$

Parameterizing the decision boundary

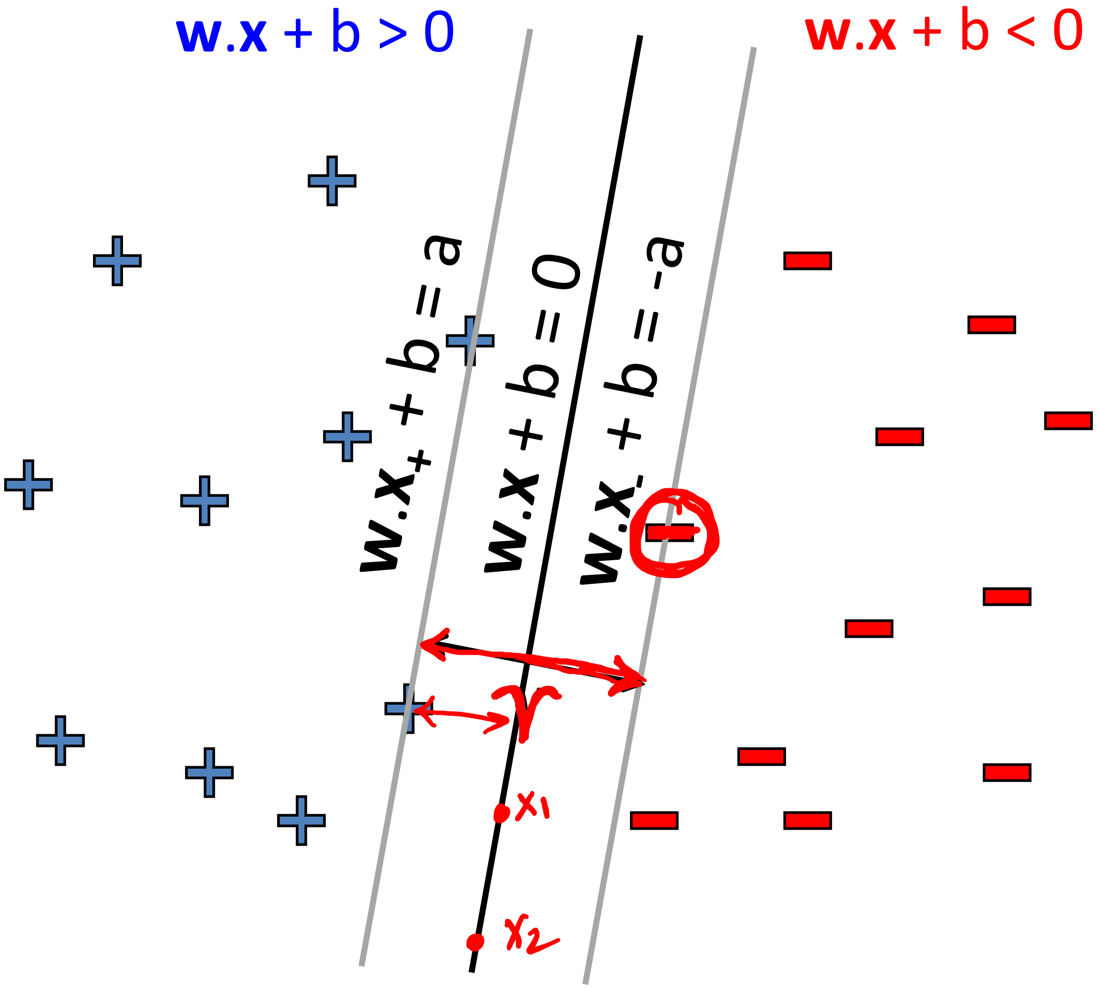


$$y_j \in \{-1, +1\} \quad \text{— class}$$

±1

$$\text{“confidence”} = \left(\mathbf{w}.\mathbf{x}_j + b\right) y_j$$

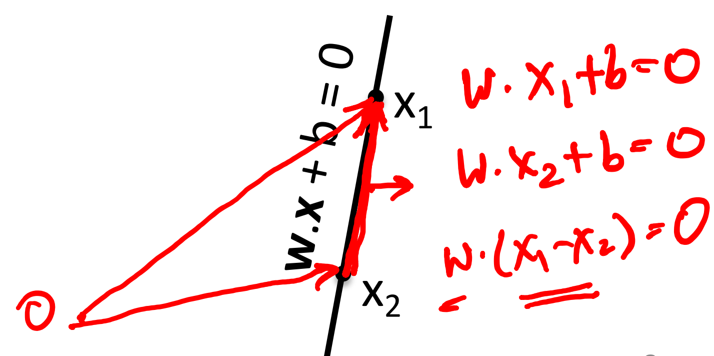
Maximizing the margin



Distance of closest examples from the line/hyperplane

$$\boxed{\text{margin} = \gamma = 2a/\|w\|}$$

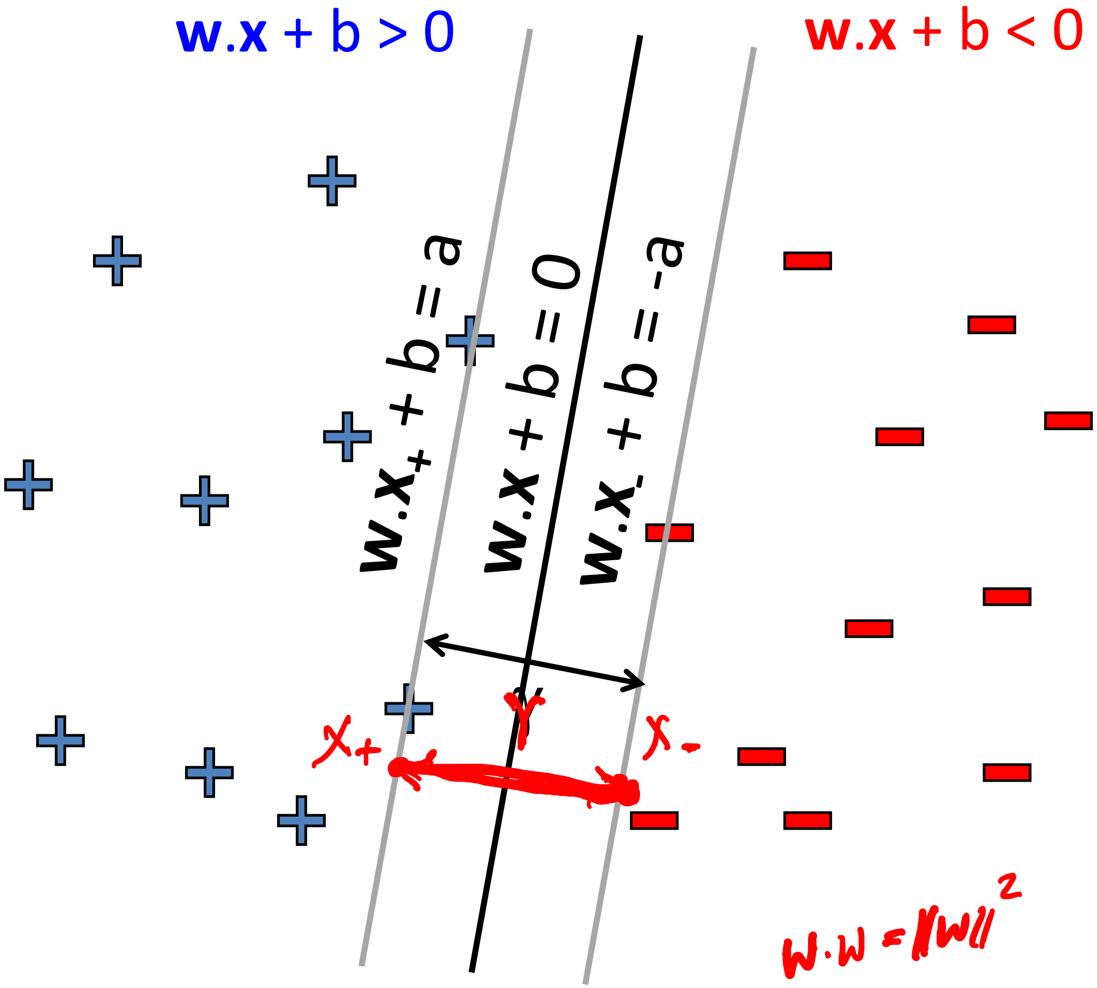
Step 1: **w** is perpendicular to lines since for any x_1, x_2 on line $\mathbf{w}.(\mathbf{x}_1 - \mathbf{x}_2) = 0$



Maximizing the margin

$\gamma = 2a/\|w\|$

$$\boxed{\text{margin} = \gamma = 2a/\|w\|}$$



Step1: **w** is perpendicular to lines

Step 2: Take a point x_- on $w.x_- + b = -a$ and move to point x_+ that is γ away on line $w.x_+ + b = a$

$$\mathbf{x}_+ = \mathbf{x}_- + \gamma\mathbf{w}/\|w\|$$

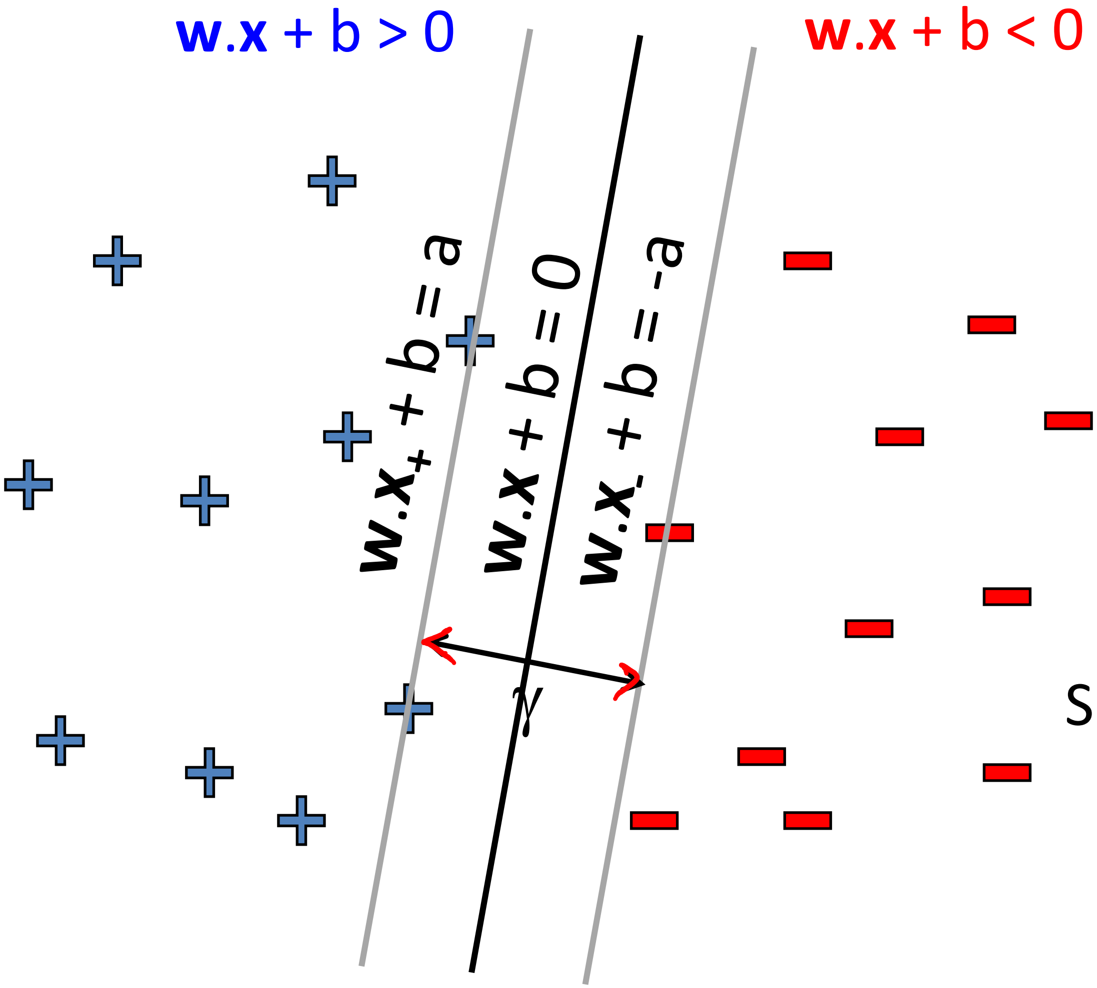
$$\mathbf{w.x}_+ = \mathbf{w.x}_- + \gamma\mathbf{w.\ w}/\|w\|$$

$$a - \cancel{b} = -a - \cancel{b} + \gamma\|w\|$$

$$2a = \gamma\|w\|$$

$2a = \gamma/\|w\|$

Maximizing the margin



Distance of closest examples from the line/hyperplane

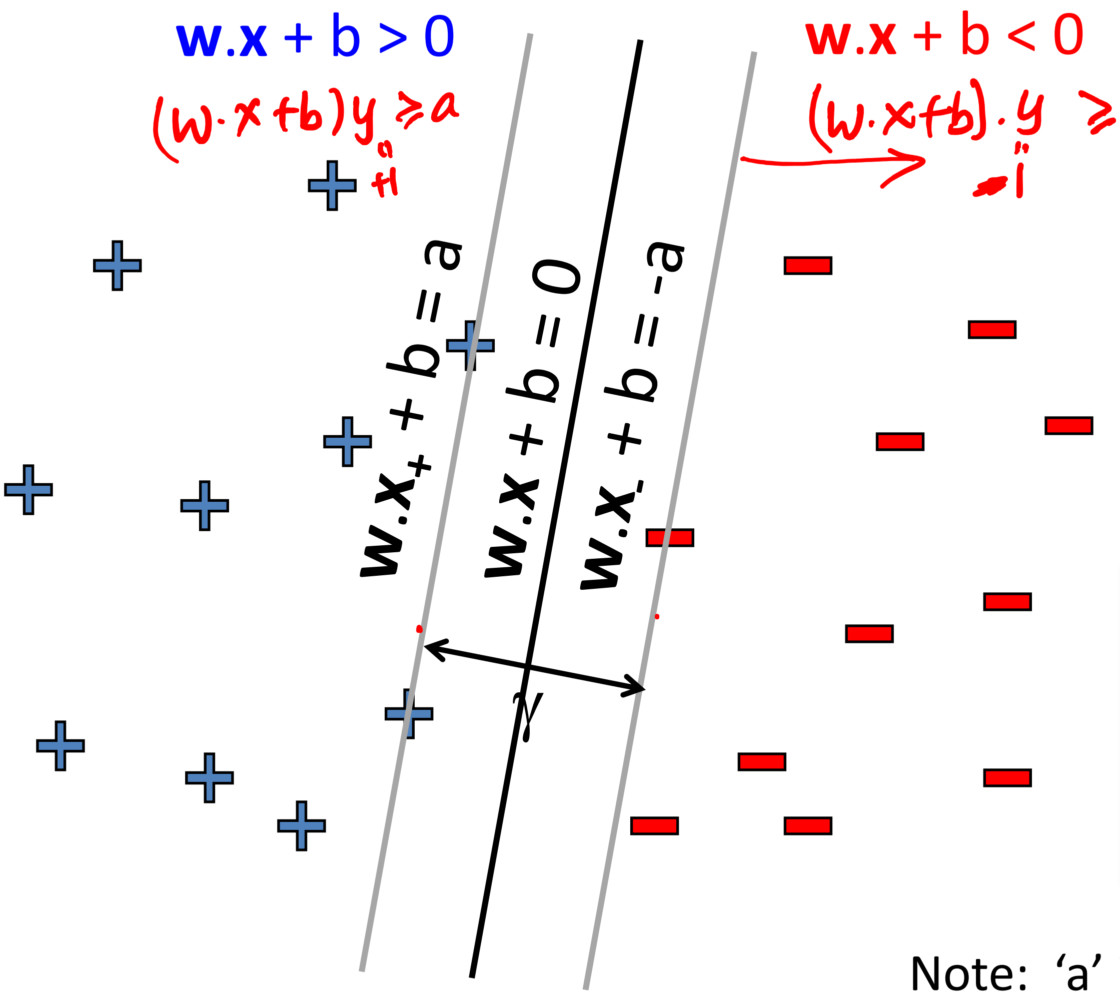
$$\boxed{\text{margin} = \gamma = 2a/\|w\|}$$

Smaller margin ⇔ larger ‖w‖

Maximizing the margin

$w \cdot x + b = 0$

$\frac{w}{a} \cdot x + \frac{b}{a} = 0$



Distance of closest examples from the line/hyperplane

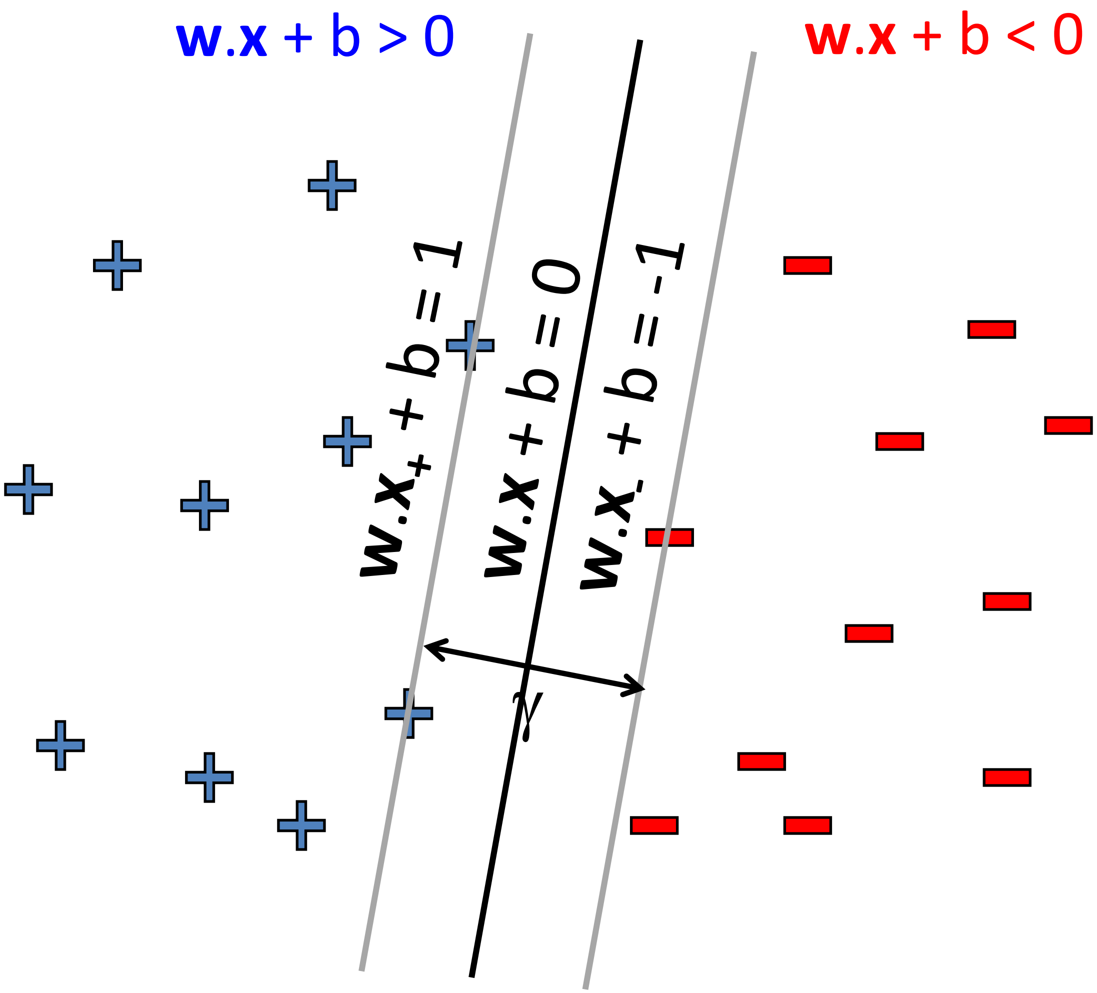
$$\boxed{\text{margin} = \gamma = 2a/\|w\|}$$

$$\max_{\mathbf{w},b} \ \gamma = 2\cancel{a}^{\,1}/\|w\|$$

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

<u>Note:</u> 'a' is arbitrary (can normalize equations by a)

Support Vector Machines



$$\min_{\mathbf{w},b} \ \mathbf{w}.\mathbf{w}$$

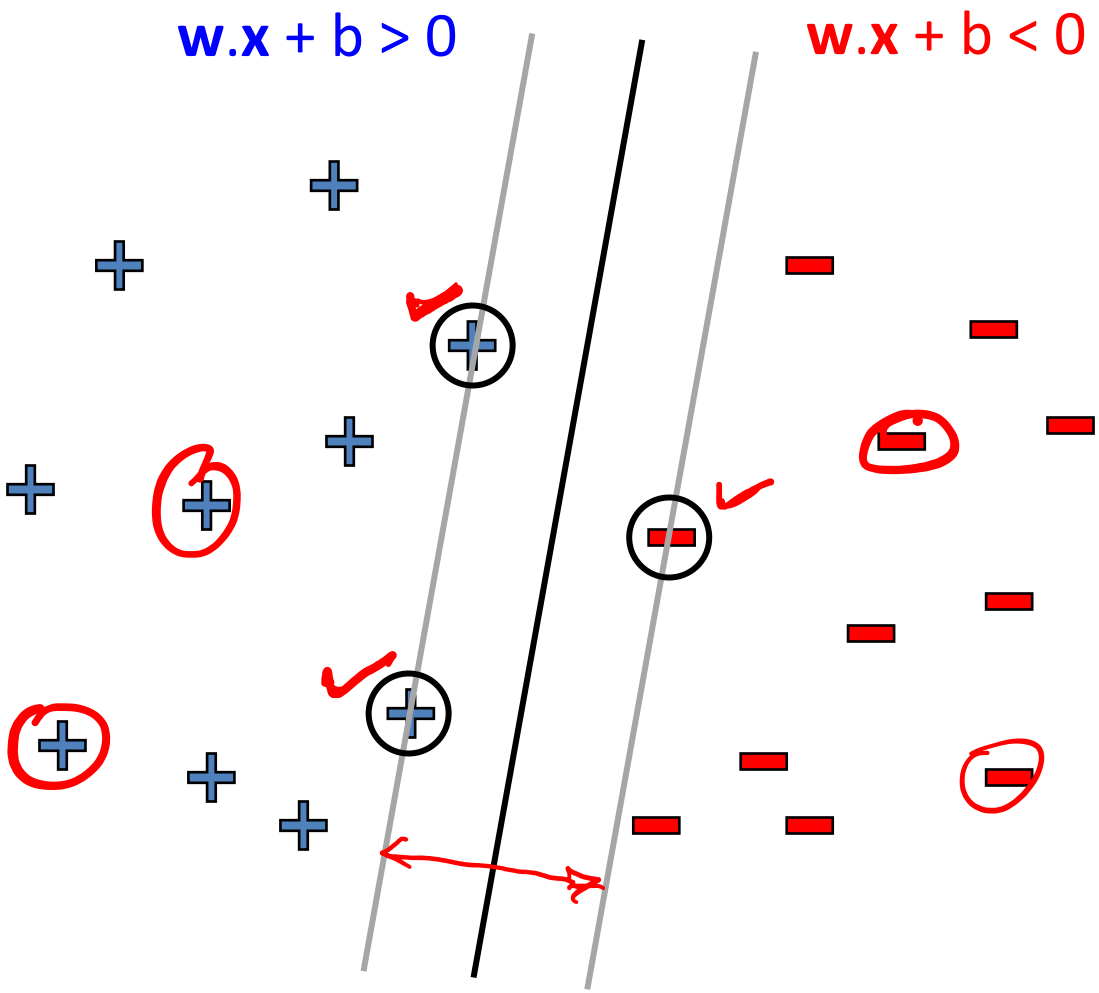
$\|w\|^2$

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

Solve efficiently by quadratic programming (QP)

- Quadratic objective, linear constraints
- Well-studied solution algorithms

Support Vectors



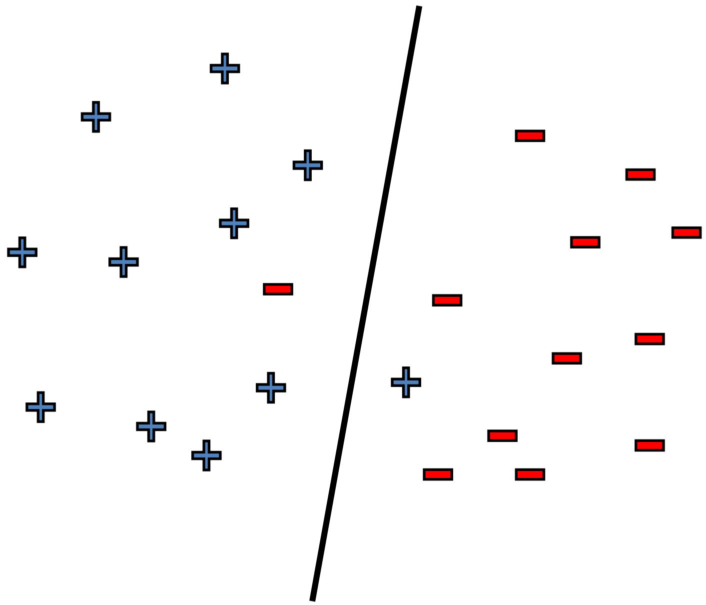
Linear hyperplane defined by "support vectors"

Moving other points a little doesn't effect the decision boundary

only need to store the support vectors to predict labels of new points

For support vectors
$(\mathbf{w}.\mathbf{x}_j+b)\ y_j = 1$

What if data is not linearly separable?



Use features of features of features of features….

x_1^2, x_2^2, x_1x_2, …., $\exp(x_1)$

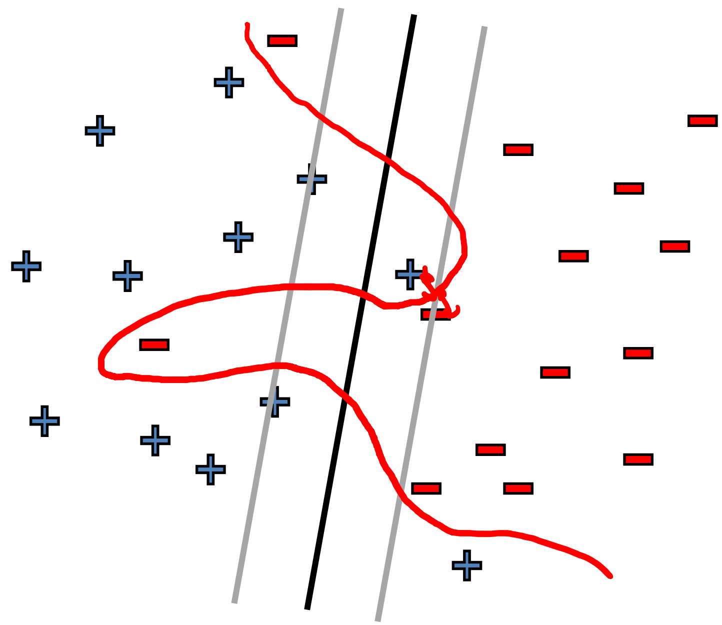
But run risk of overfitting!

What if data is still not linearly separable?

$\sum_j \mathbb{1}_{(w\cdot x_j+b)\, y_j > 0}$

Allow "error" in classification

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



Smaller margin ⇔ larger ‖w‖

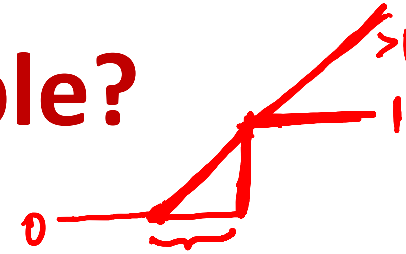
Maximize margin and minimize # mistakes on training data

C - tradeoff parameter

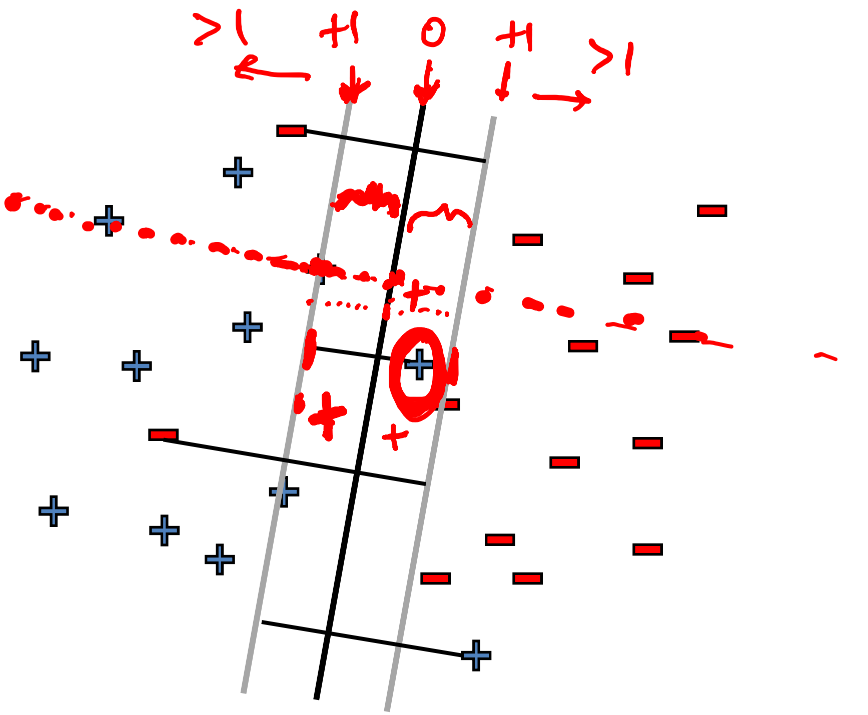
Not QP ☹

0/1 loss (doesn't distinguish between near miss and bad mistake)

What if data is still not linearly separable?



Allow "error" in classification



Soft margin approach

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

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

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

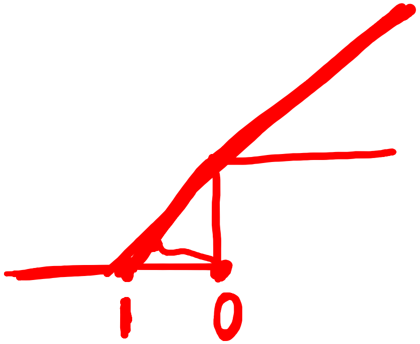
ξ_j - "slack" variables

= (>1 if x_j misclassifed)

pay linear penalty if mistake

C - tradeoff parameter (chosen by cross-validation)

Still QP ☺

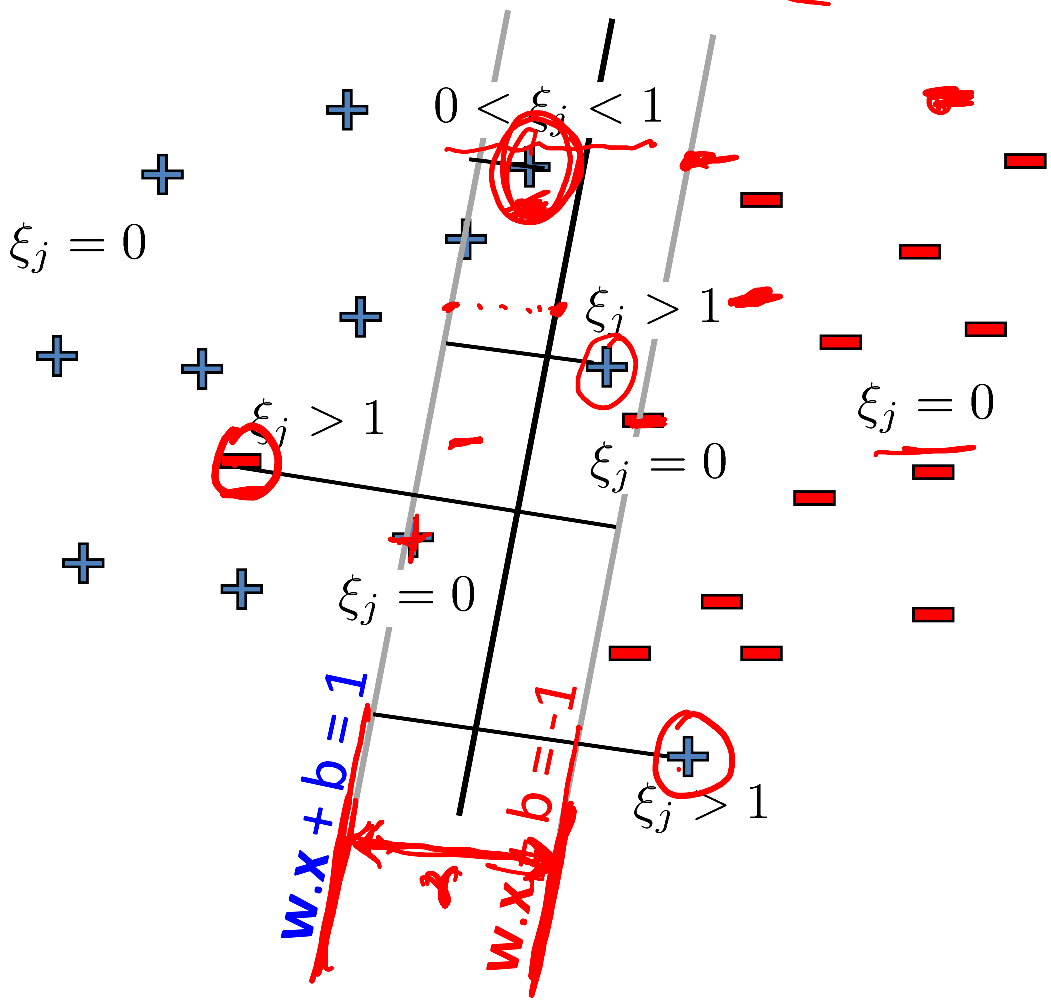


Soft-margin SVM

$(w \cdot x + b) y \geq 1$

$1 - \xi_j < 0$

$\xi_j > 1$



Soften the constraints:

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

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

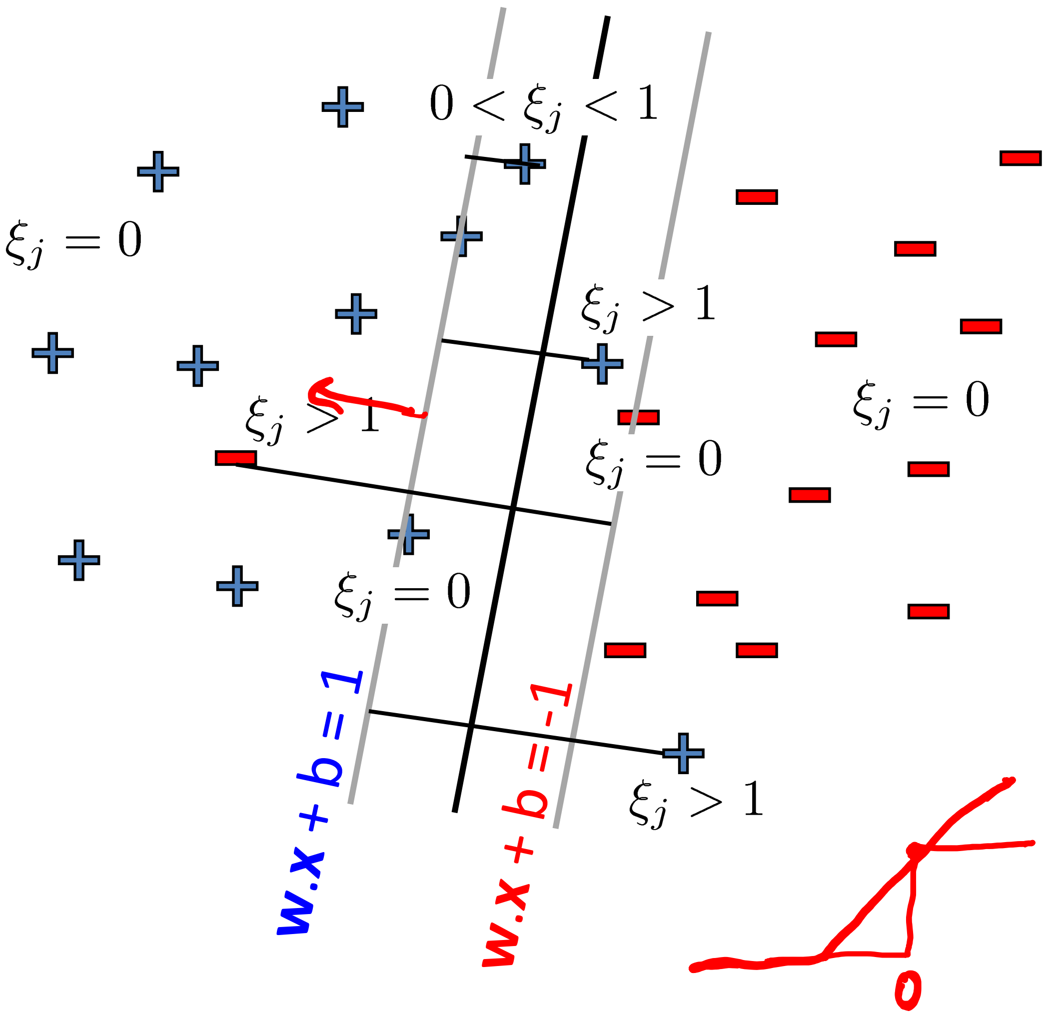
Penalty for misclassifying:

$$C\, \xi_j$$

How do we recover hard margin SVM?

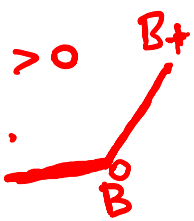
Set $C = \infty$

Slack variables – Hinge loss



$\|w\|^2 + C\sum_j \xi_j$

$B_+ = \begin{cases} B & \text{if } B > 0 \\ 0 & \text{o.w.} \end{cases}$



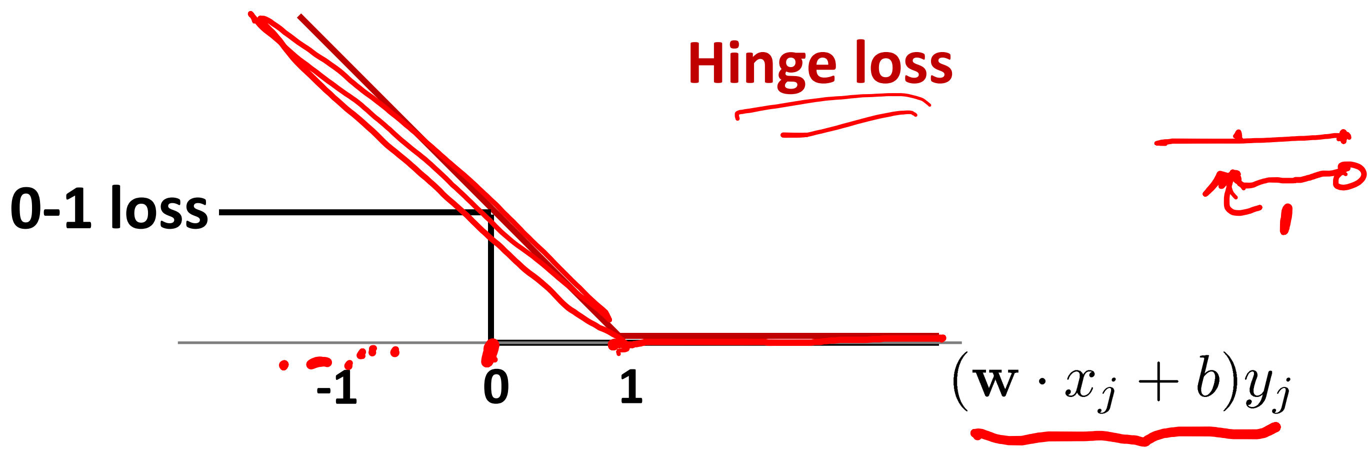
Notice that

$$\xi_j = (1 - \underbrace{(\mathbf{w} \cdot x_j + b) y_j}_{\text{confidence}})_+$$

$$= \begin{cases} 1 & = 0 \\ 1 + \dots & < 0 \\ 0 & > 1 \end{cases}$$

Slack variables – Hinge loss

$$\xi_j = (1 - (\mathbf{w} \cdot x_j + b)y_j))_+$$



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

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

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

$\Leftrightarrow$

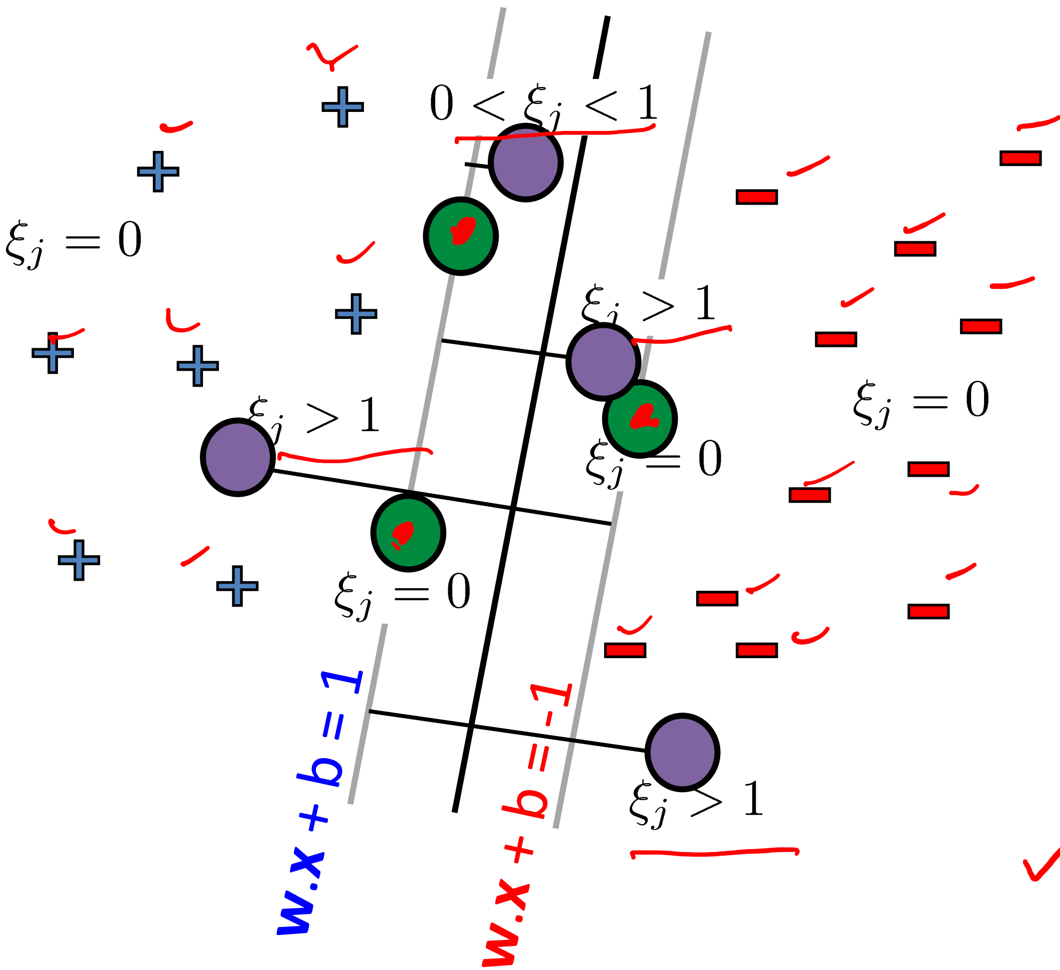
Regularized hinge loss

$$\min_{\mathbf{w},b} \ \mathbf{w.w} + C \underbrace{\sum_j (1-(\mathbf{w}.x_j+b)y_j)_+}_{\xi_j}$$

regularization parameter

Support Vectors

any training points that affect decision boundary (w, b)



Margin support vectors

$\xi_j = 0$, $(\mathbf{w}.\mathbf{x}_j + b)\, y_j = 1$

(don't contribute to objective but enforce constraints on solution)

Correctly classified but on margin

Non-margin support vectors

$\xi_j > 0$ $(w \cdot x_j + b) y_j \leq 1$

(contribute to both objective and constraints)

$1 > \xi_j > 0$ Correctly classified but inside margin

$\xi_j > 1$ Incorrectly classified

SVM vs. Logistic Regression

$\|w\|^2 + C$ hingeloss

SVM : **Hinge loss**

$$\text{loss}(f(x_j), y_j) = (1 - (\mathbf{w} \cdot x_j + b) y_j))_+$$

Logistic Regression : **Log loss** (-ve log conditional likelihood)

$$\text{loss}(f(x_j), y_j) = -\log P(y_j \mid x_j, \mathbf{w}, b) = \log(1 + e^{-(\mathbf{w} \cdot x_j + b) y_j})$$

Surrogate losses