Support Vector Machines

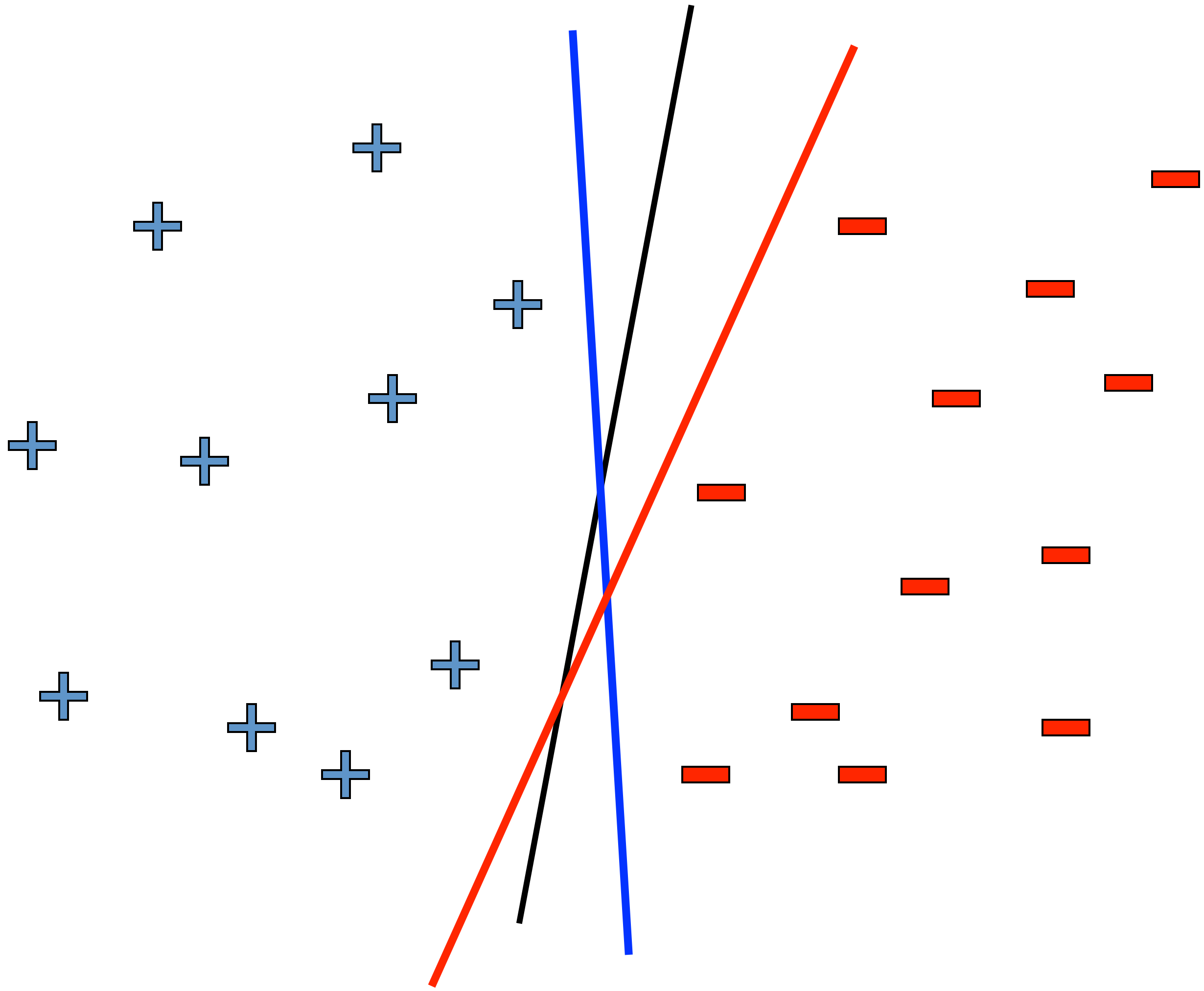
Aarti Singh and Eric Xing

Machine Learning 10-701/15-781
Oct 3, 2012

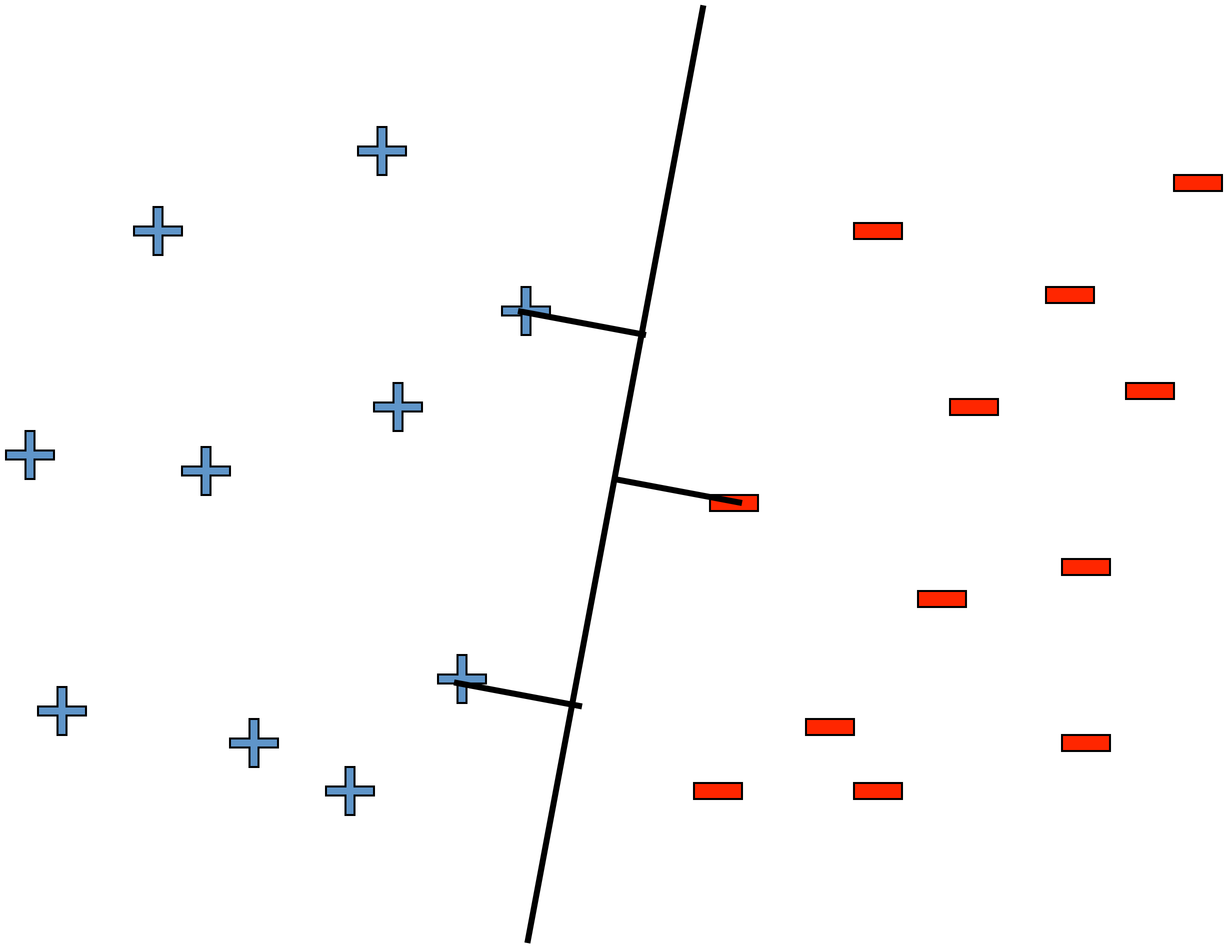
MACHINE LEARNING DEPARTMENT
Carnegie Mellon.
School of Computer Science

At Pittsburgh G-20 summit …

Linear classifiers – which line is better?



Pick the one with the largest margin!



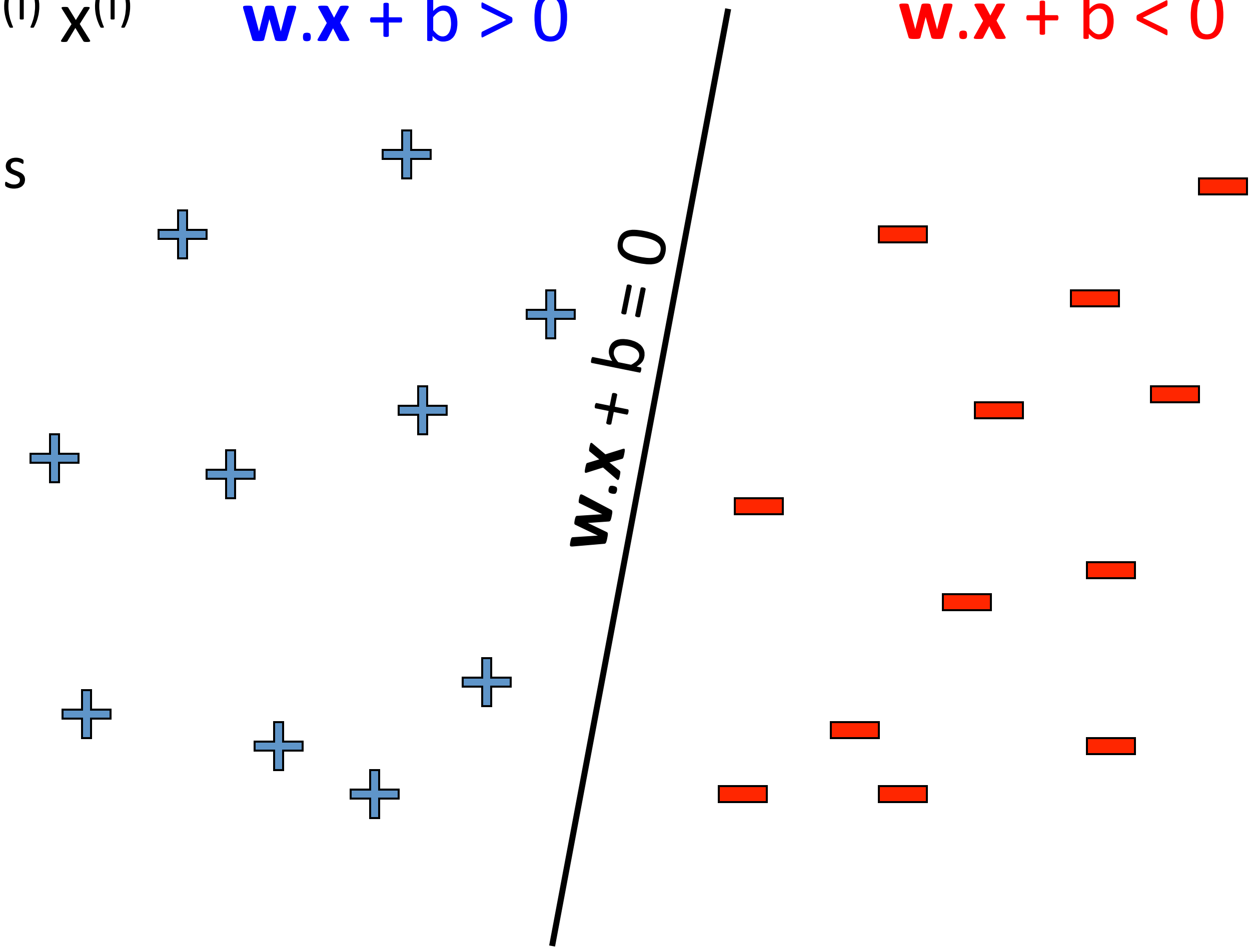
Parameterizing the decision boundary

$$\mathbf{w}.\mathbf{x} = \sum_{i=1}^{m} w^{(i)}\, x^{(i)}$$

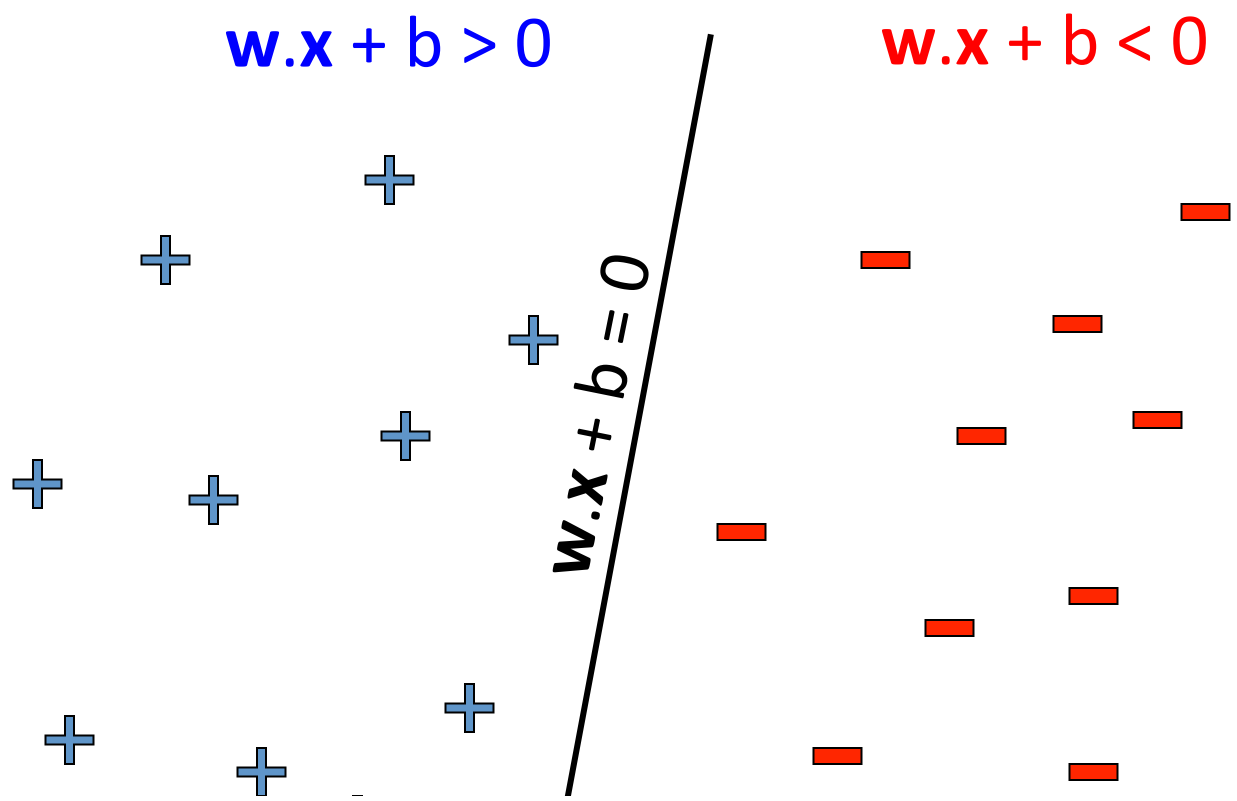
m features

$\mathbf{w}.\mathbf{x} + b > 0$

$\mathbf{w}.\mathbf{x} + b < 0$

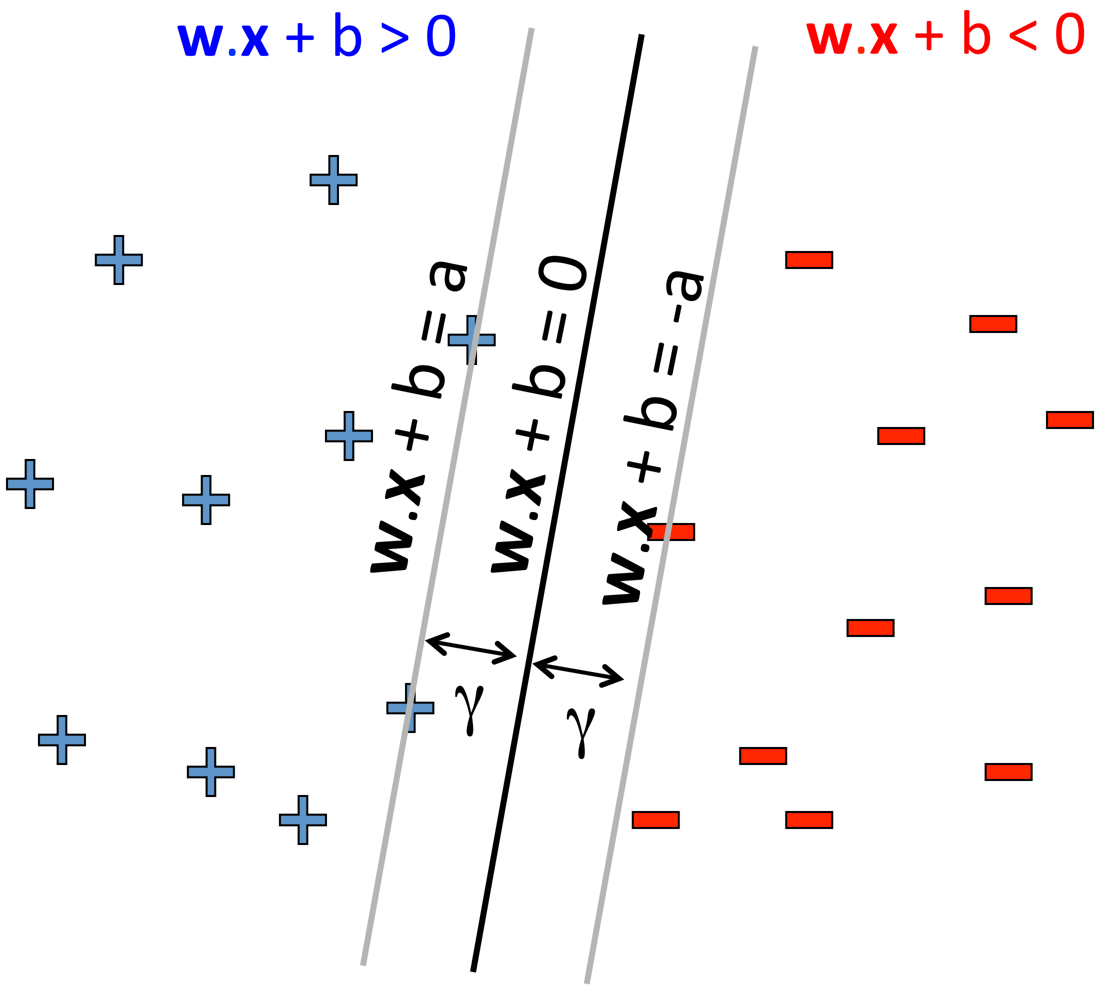


Parameterizing the decision boundary



"confidence" for j^{th} data point $= \left(\mathbf{w}.\mathbf{x}_j + b\right) y_j$

Maximizing the margin



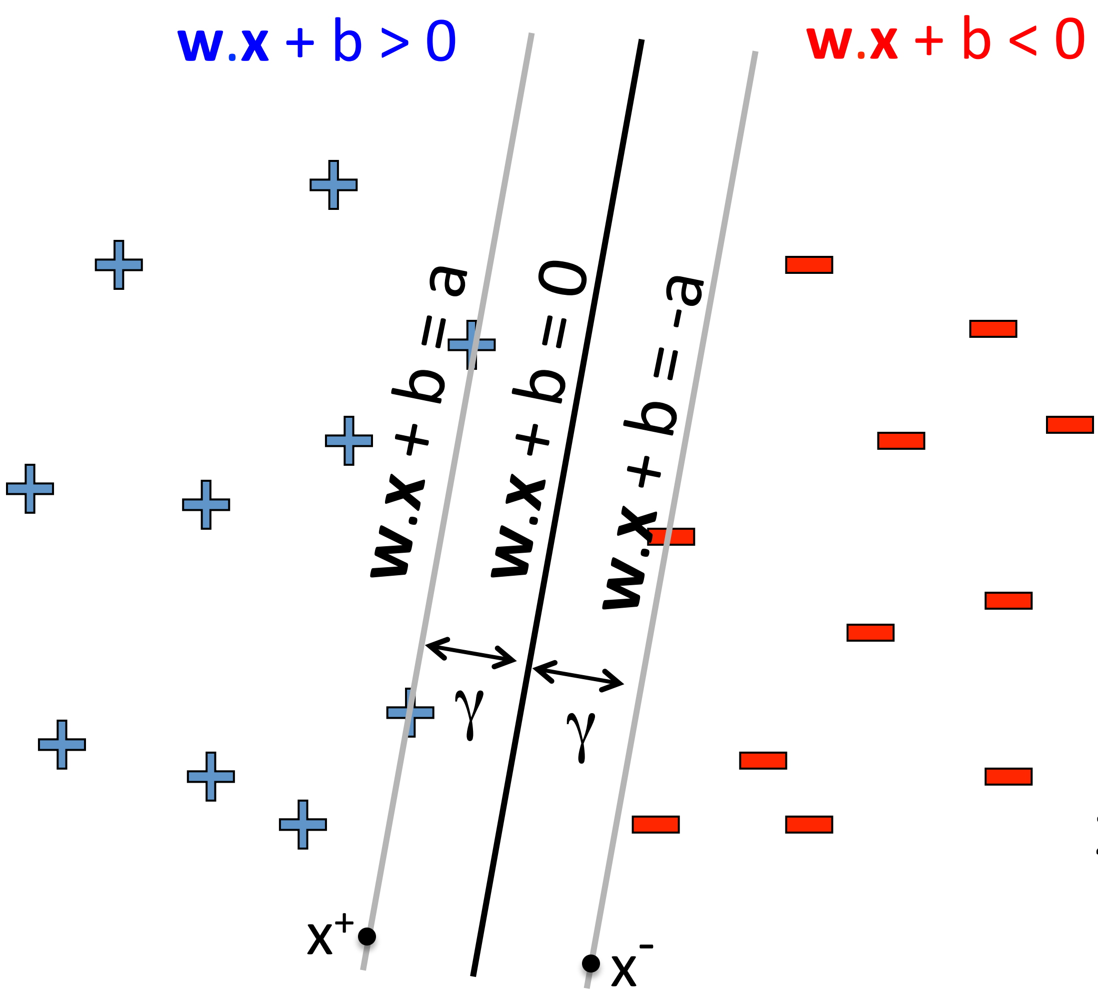
Distance between examples closest to the line/hyperplane

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

Proof sketch:

1) **w** is perpendicular to line
   $\langle \mathbf{w}, x_1 - x_2 \rangle = 0$ if x_1, x_2 on line

Maximizing the margin



Distance between examples closest to the line/hyperplane

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

Proof sketch:

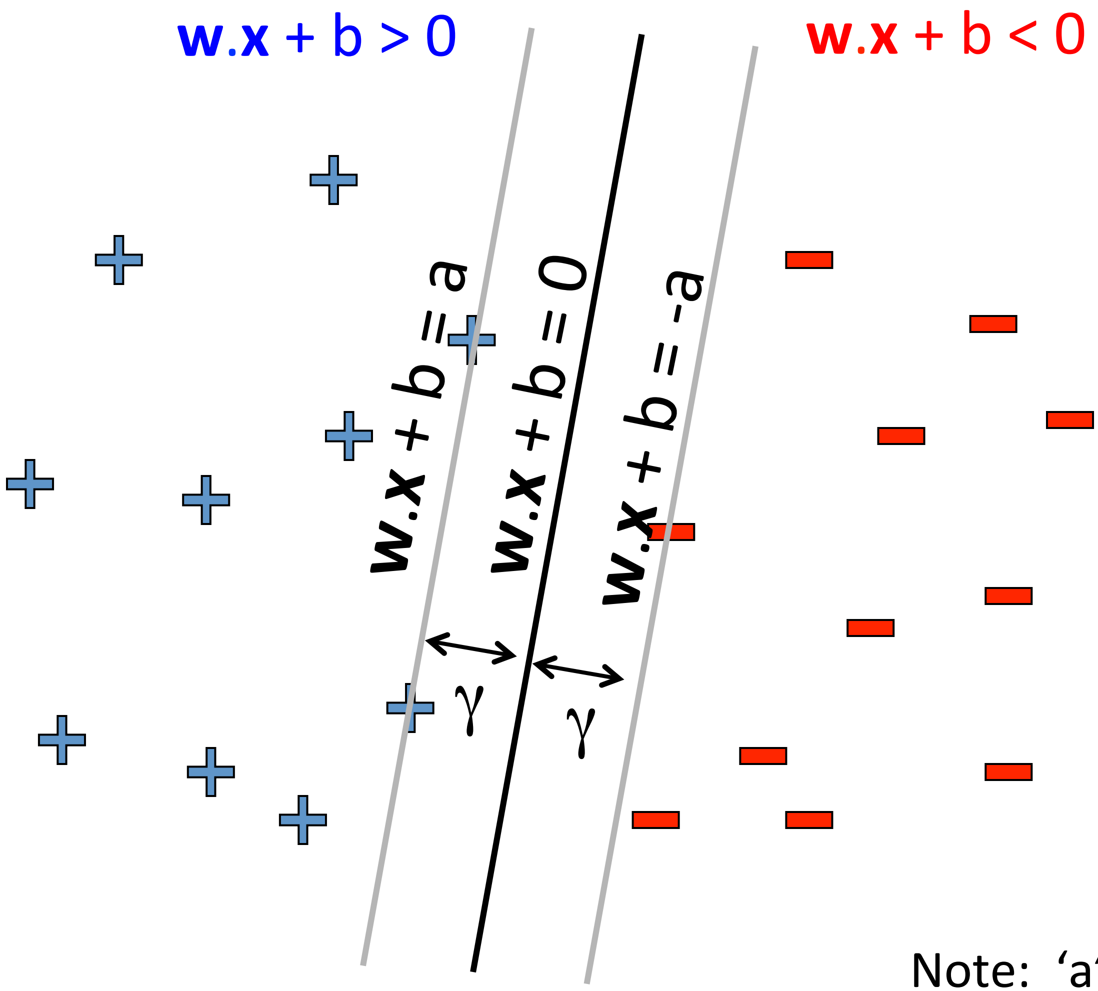
1) **w** is perpendicular to line
   $\langle \mathbf{w}, x_1 - x_2 \rangle = 0$ if x_1, x_2 on line

2) $x^+ = x^- + 2\gamma\, \mathbf{w}/\|w\|$

$\Rightarrow a = -a + 2\gamma\, \mathbf{w}.\mathbf{w}/\|w\|$

since $\mathbf{w}.x^+ + b = a$, $\mathbf{w}.x^- + b = -a$

Maximizing the margin



Distance between examples closest to the line/hyperplane

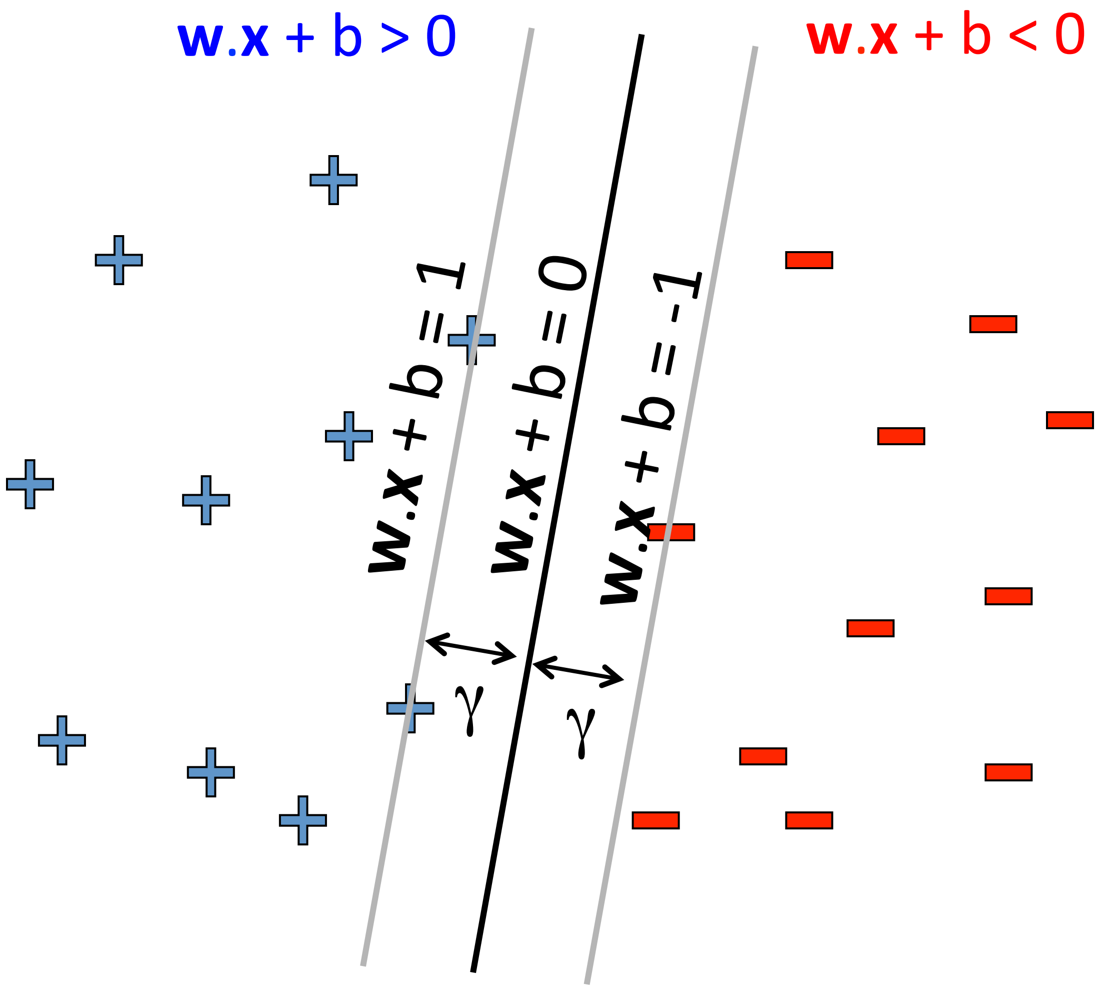
margin = $2\gamma = 2a/\|w\|$

$$\max_{\mathbf{w},b} \; 2\gamma = 2a/\|w\|$$

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

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

Support Vector Machines



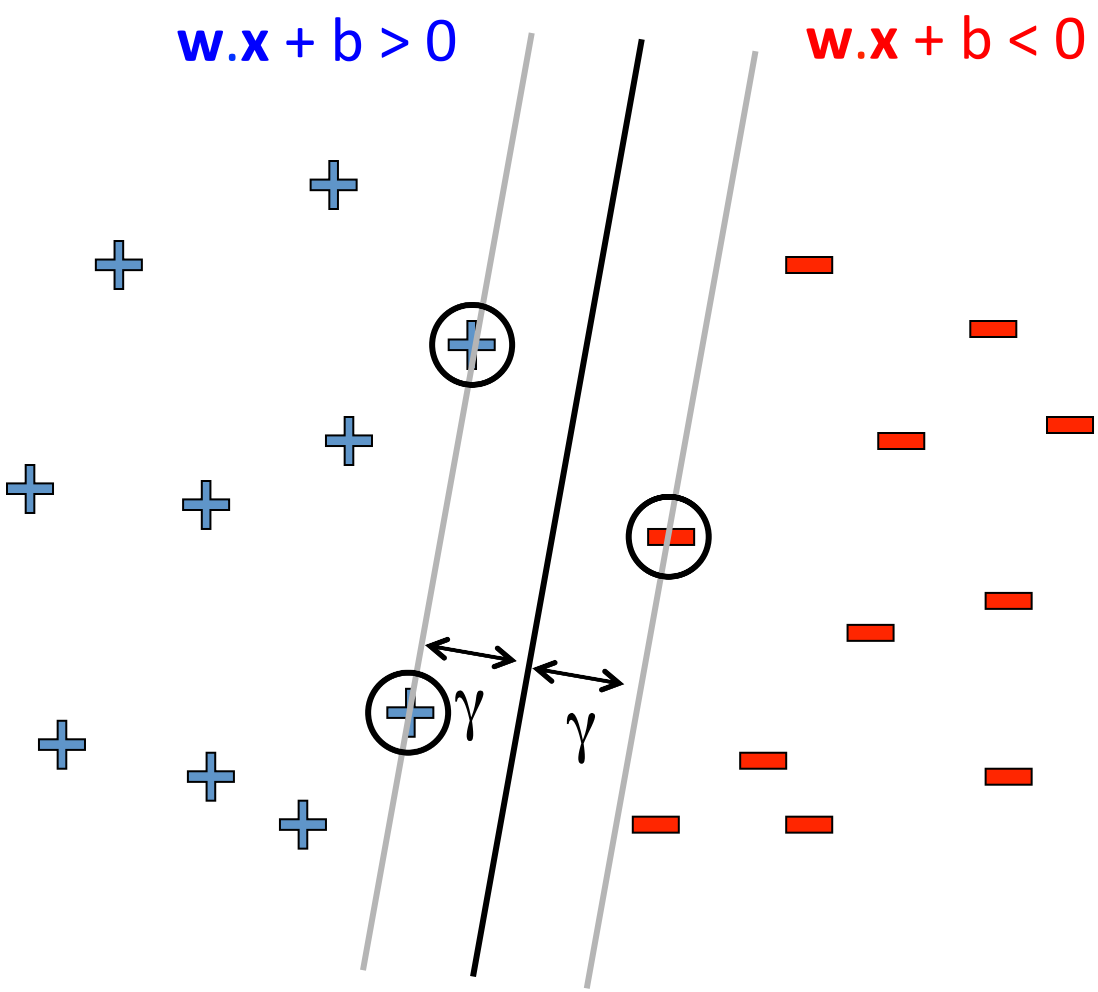
$$\min_{\mathbf{w},b} \; \mathbf{w}.\mathbf{w}$$

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

Solve efficiently by quadratic programming (QP)

- Well-studied solution algorithms

Support Vectors



Linear hyperplane defined by "<u>support vectors</u>"

$$i: (\mathbf{w}.\mathbf{x}_i + b)\, y_i = 1$$

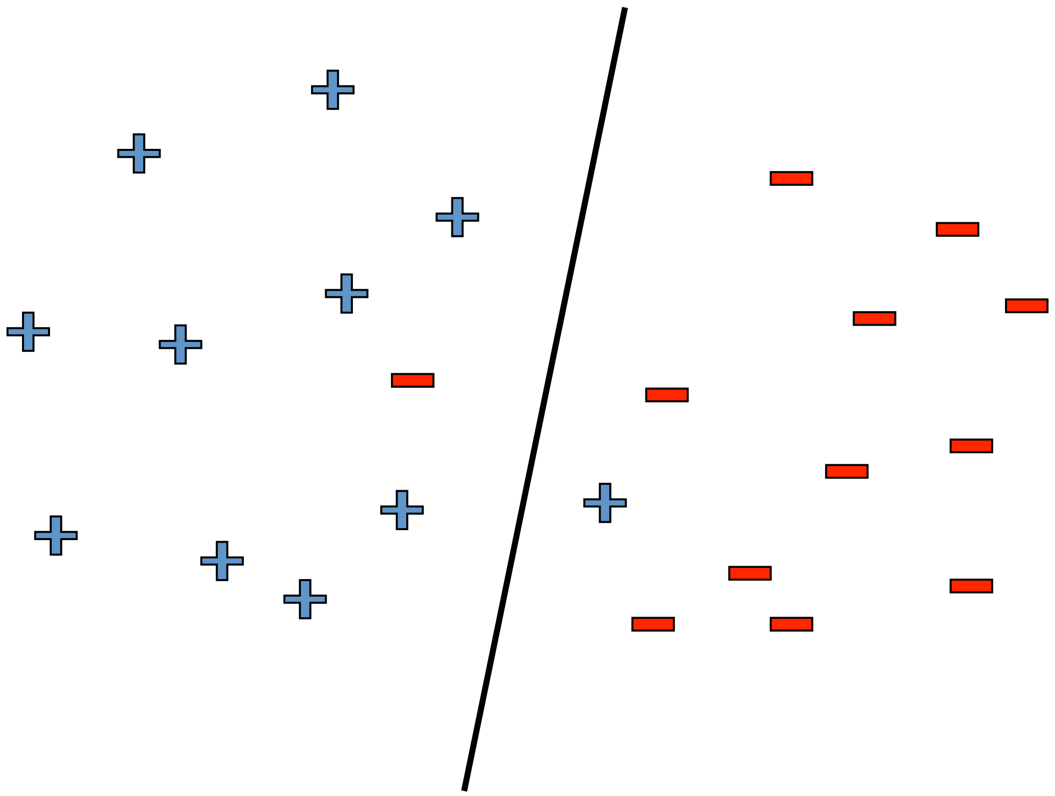
Moving other points a little doesn't effect the decision boundary

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

How many support vectors in linearly separable case?

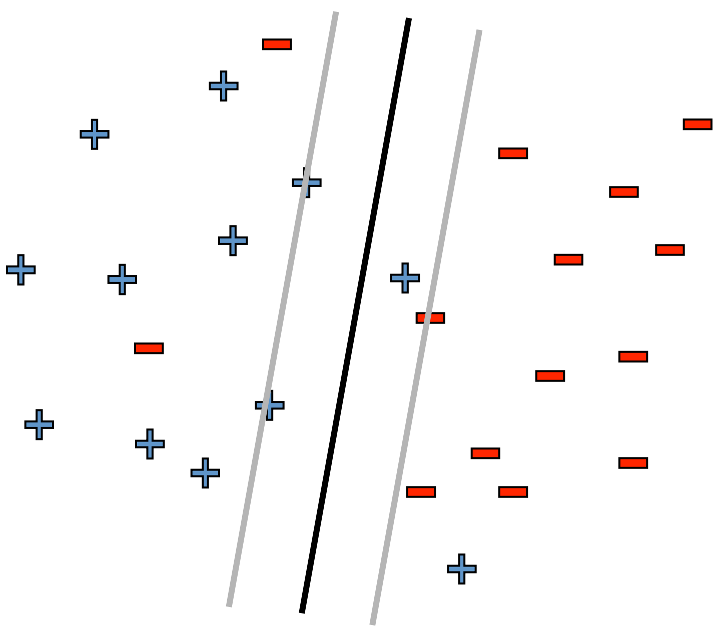
$\leq m+1$

What if data is not linearly separable?



What if data is still not linearly separable?

Allow "error" in classification



$$\min_{\mathbf{w},b} \ \mathbf{w}.\mathbf{w} + C \ \#\text{mistakes}$$

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

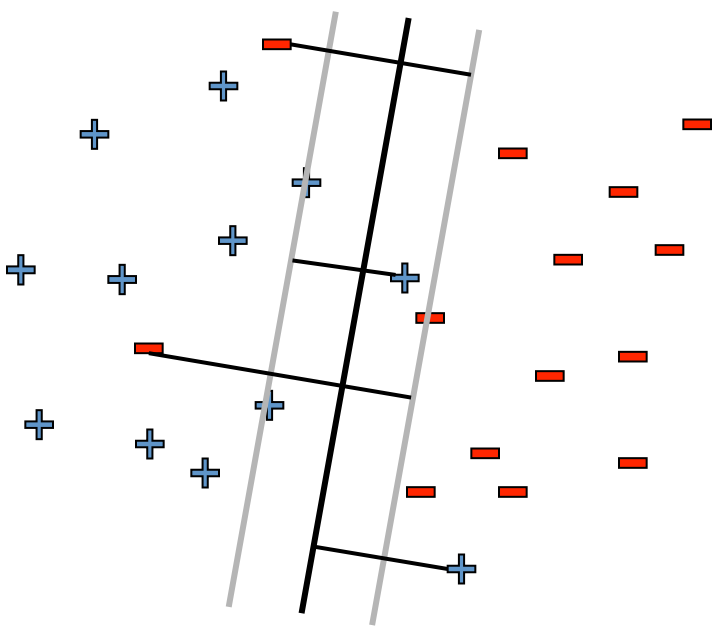
Maximize margin and minimize # mistakes on training data

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

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

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

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

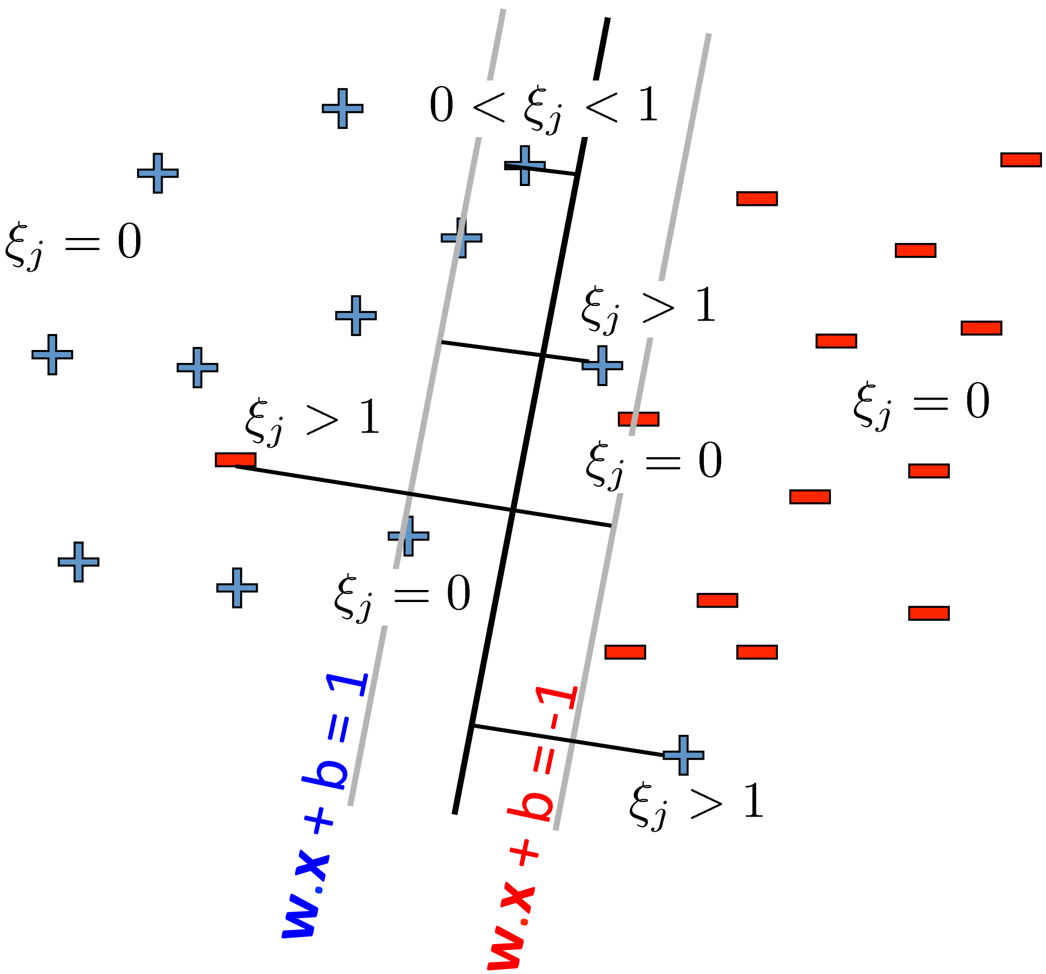
ξ_j - "slack" variables (>1 if x_j misclassifed)

pay linear penalty if mistake

C - tradeoff parameter (chosen by cross-validation)

Still QP ☺

Soft-margin SVM



Soften the constraints:

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

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

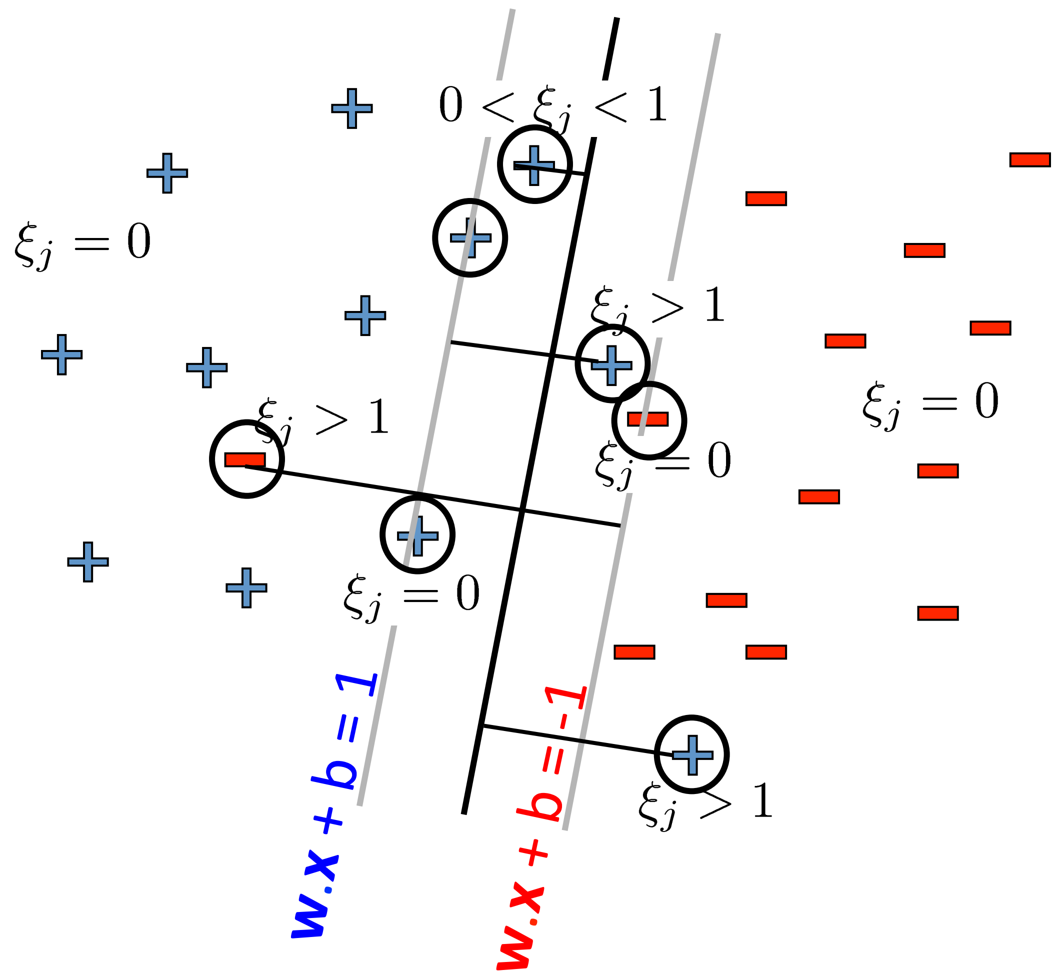
Penalty for misclassifying:

$$C\, \xi_j$$

How do we recover hard margin SVM?

Set $C = \infty$

Support Vectors



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 = ∞

Slack variables as Hinge loss

Regularized loss

$$\xi_j = \mathrm{loss}(f(x_j), y_j)$$

$$f(x_j) = \mathrm{sgn}(\mathbf{w} \cdot \mathrm{x_j} + \mathrm{b})$$

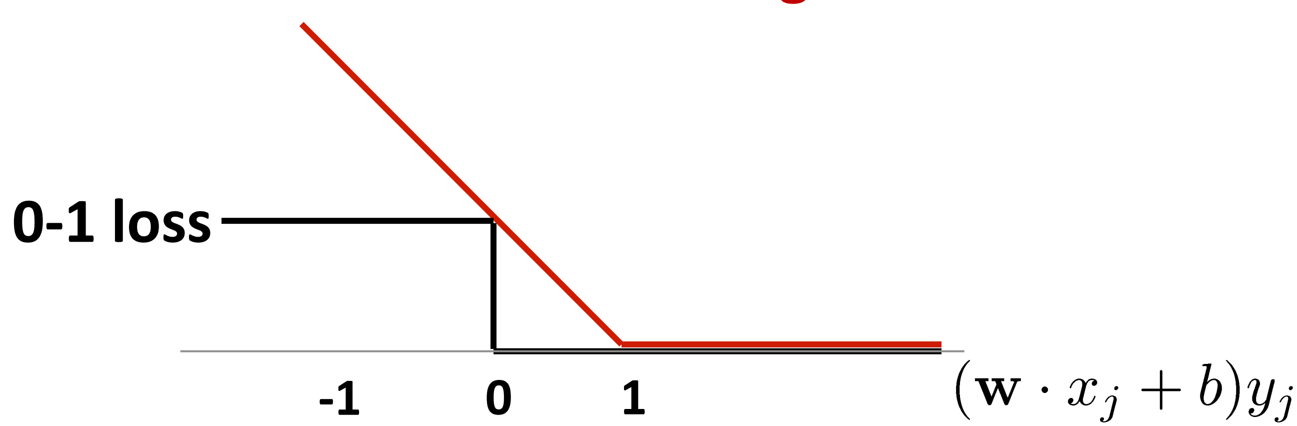
$$\min_{\mathbf{w}, b, \xi_j} \ \mathbf{w}.\mathbf{w} + C \sum_j \xi_j$$

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

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

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

Hinge loss



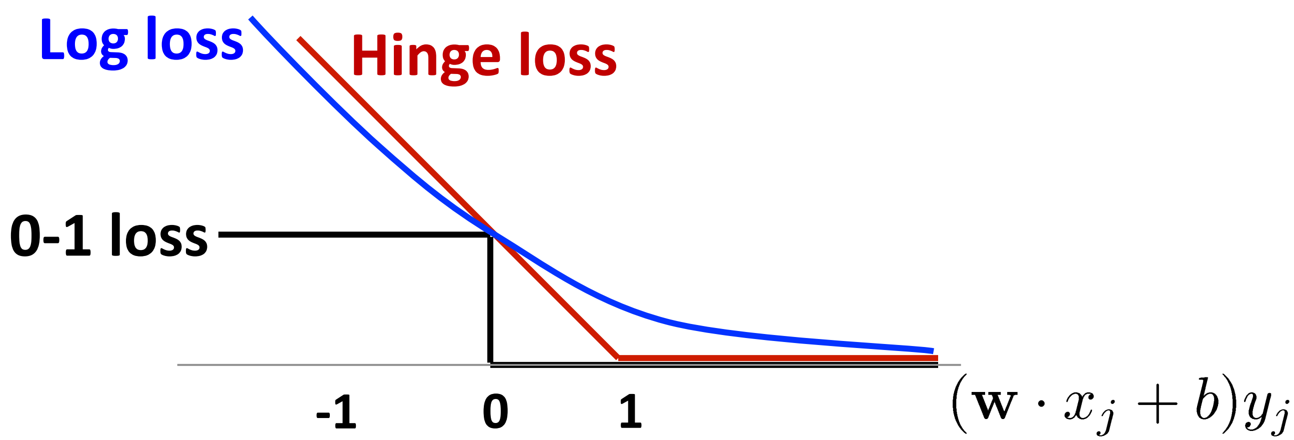
SVM vs. Logistic Regression

SVM : **Hinge loss**

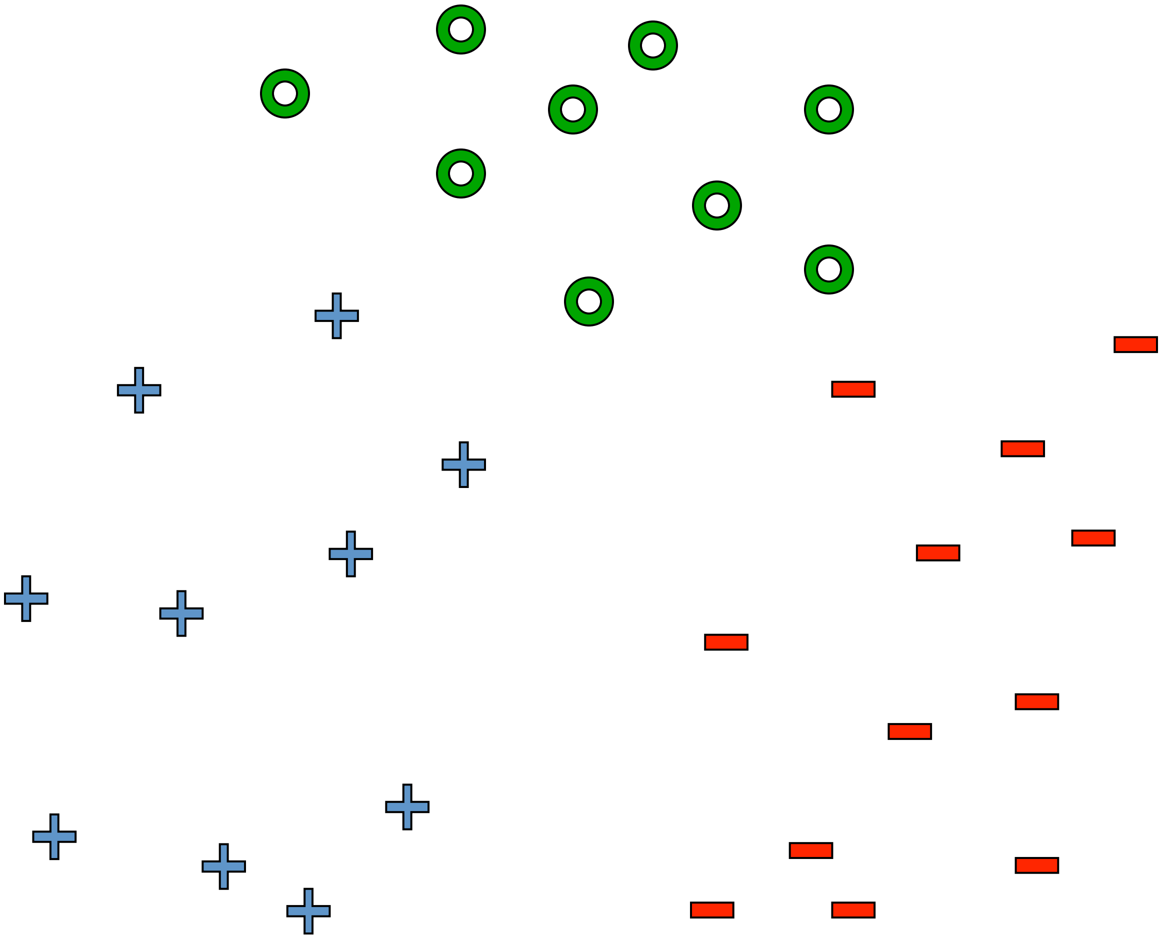
$$\text{loss}(f(x_j), y_j) = (1 - (\mathbf{w} \cdot x_j + b)y_j))_+$$

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

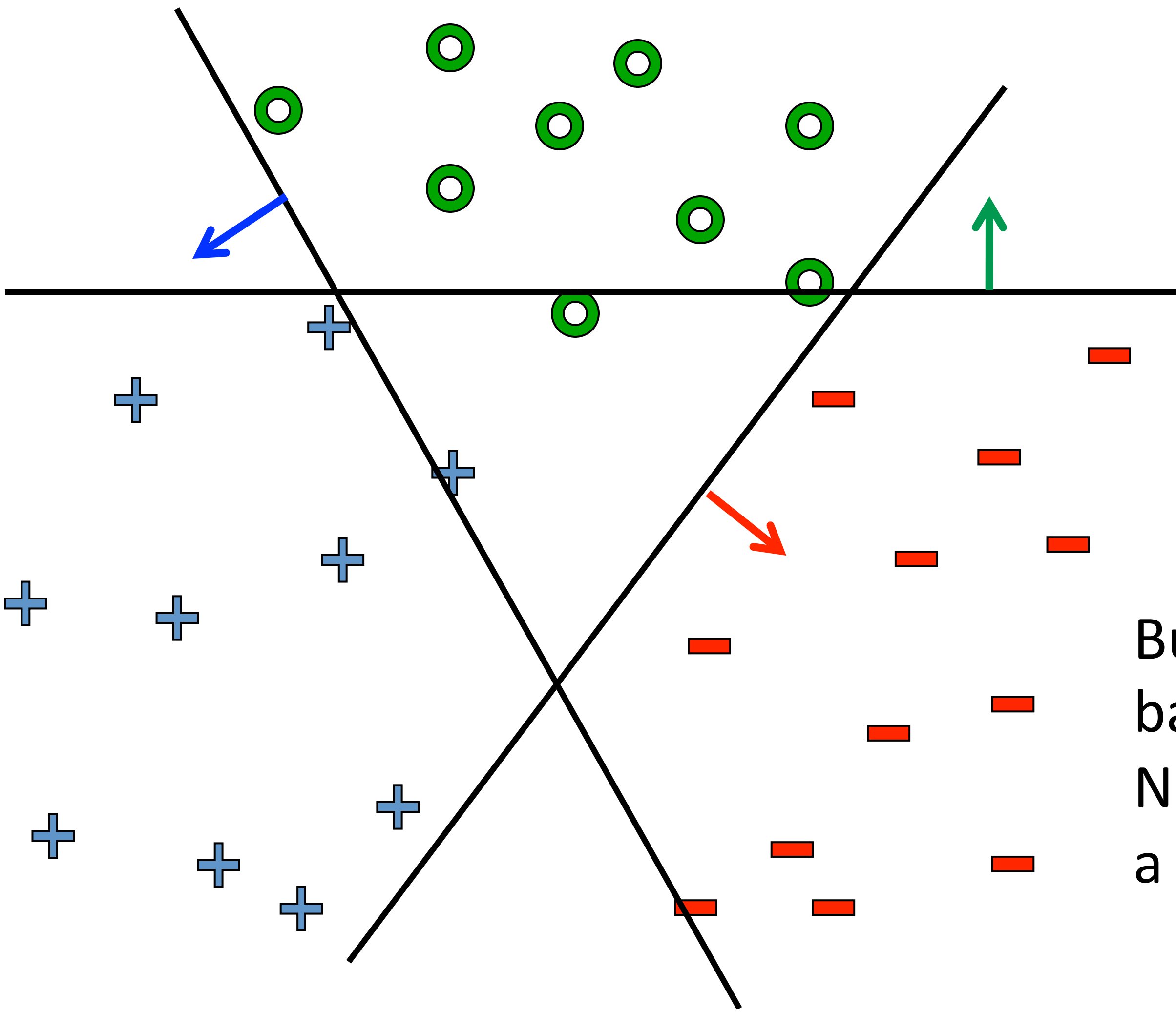
$$\text{loss}(f(x_j), y_j) = -\log P(y_j \mid x_j, \mathbf{w}, b) = \log(1 + e^{-(\mathbf{w} \cdot x_j + b)y_j})$$



What about multiple classes?



One against all



Learn 3 classifiers separately:

Class k vs. rest

$(\mathbf{w}_k, b_k)_{k=1,2,3}$

$$y = \arg\max_k \mathbf{w}_k.x + b_k$$

But $\mathbf{w}_k$s may not be based on the same scale.

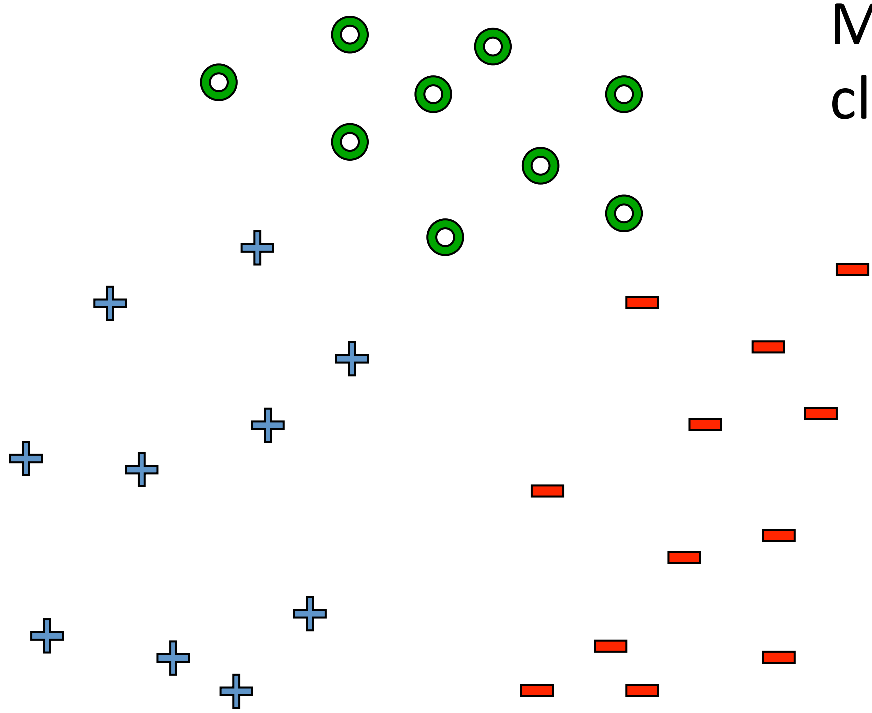
Note: $(a\mathbf{w}).x + (ab)$ is also a solution

Learn 1 classifier: Multi-class SVM

Simultaneously learn 3 sets of weights

$$\text{minimize}_{\mathbf{w},b} \quad \sum_y \mathbf{w}^{(y)}.\mathbf{w}^{(y)}$$

$$\text{s.t.} \;\; \mathbf{w}^{(y_j)}.\mathbf{x}_j + b^{(y_j)} \geq \mathbf{w}^{(y')}.\mathbf{x}_j + b^{(y')} + 1, \;\; \forall y' \neq y_j, \;\; \forall j$$



Margin - gap between correct class and nearest other class

y = arg max $\mathbf{w}^{(k)}.x + b^{(k)}$

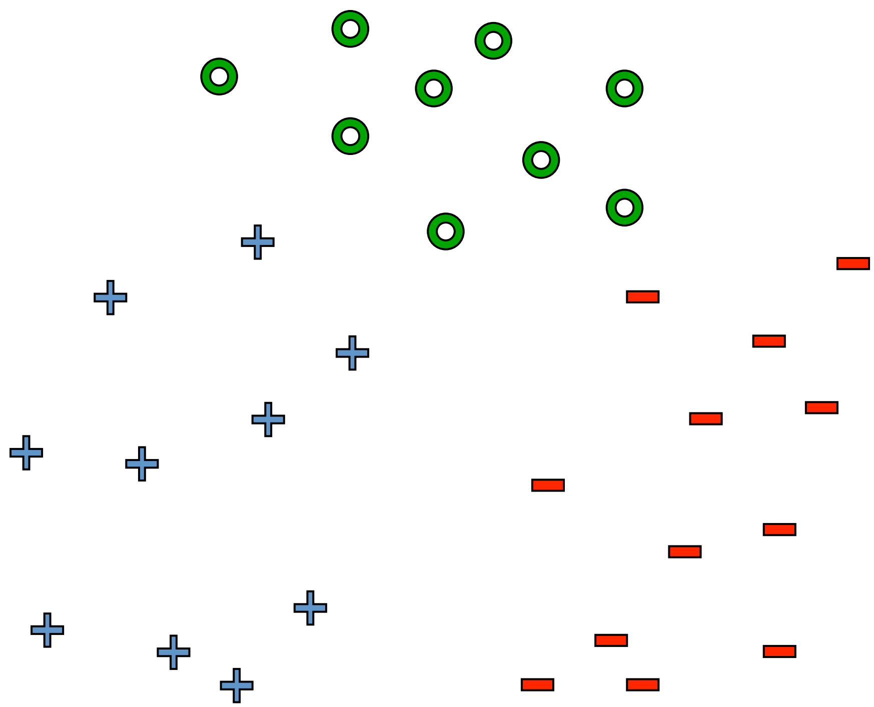
Learn 1 classifier: Multi-class SVM

Simultaneously learn 3 sets of weights

$$\text{minimize}_{\mathbf{w},b} \quad \sum_y \mathbf{w}^{(y)}.\mathbf{w}^{(y)} + C \sum_j \sum_{y \neq y_j} \xi_j^{(y)}$$

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

$$\xi_j^{(y)} \geq 0 \quad , \ \forall y \neq y_j, \ \forall j$$



y = arg max $\mathbf{w}^{(k)}.x + b^{(k)}$

Joint optimization: $\mathbf{w}_k$s have the same scale.

What you need to know

- Maximizing margin
- Derivation of SVM formulation
- Slack variables and hinge loss
- Relationship between SVMs and logistic regression
  - 0/1 loss
  - Hinge loss
  - Log loss
- Tackling multiple class
  - One against All
  - Multiclass SVMs