# Support Vector Machines

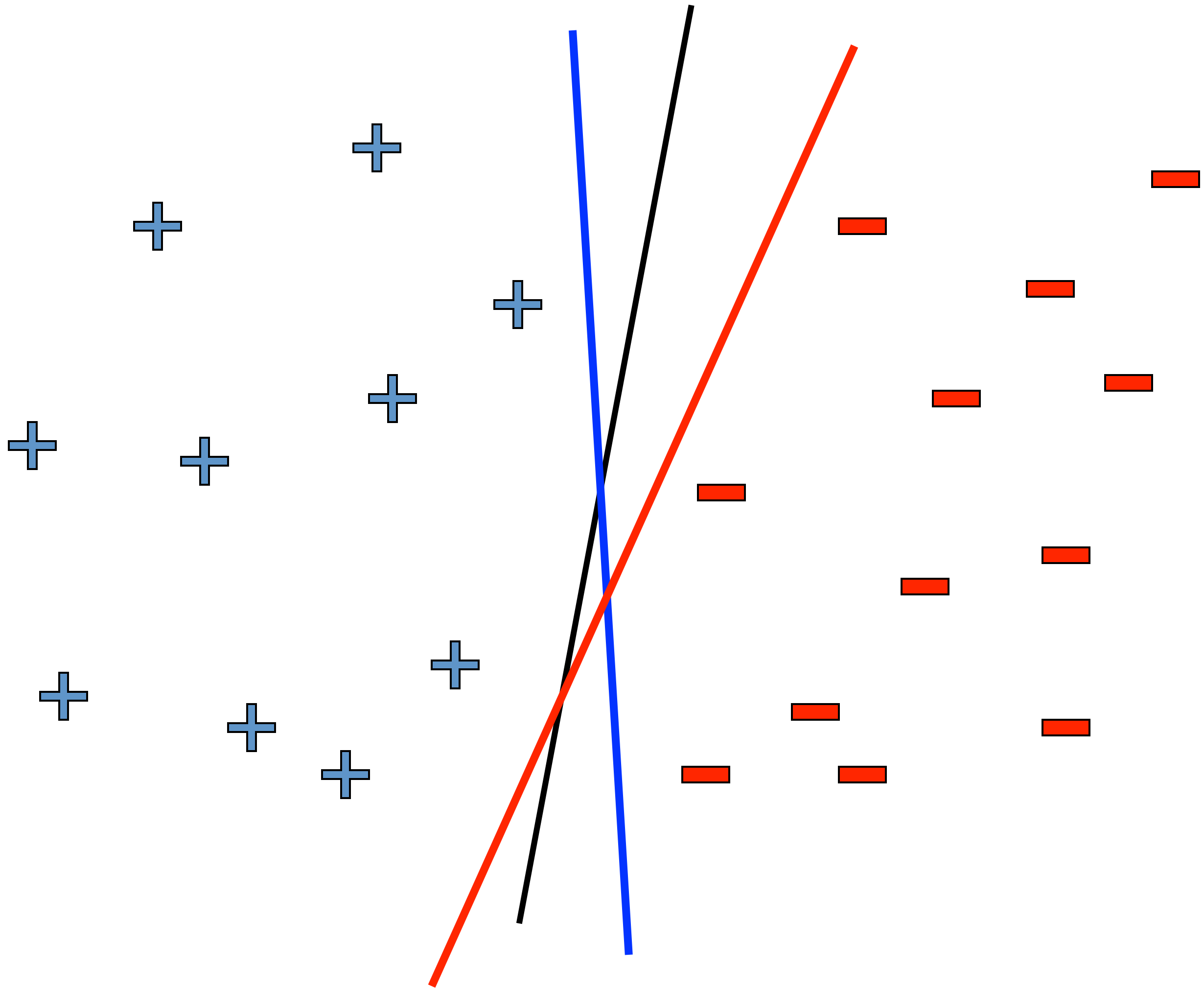
Aarti Singh

Machine Learning 10-601
Nov 17, 2011

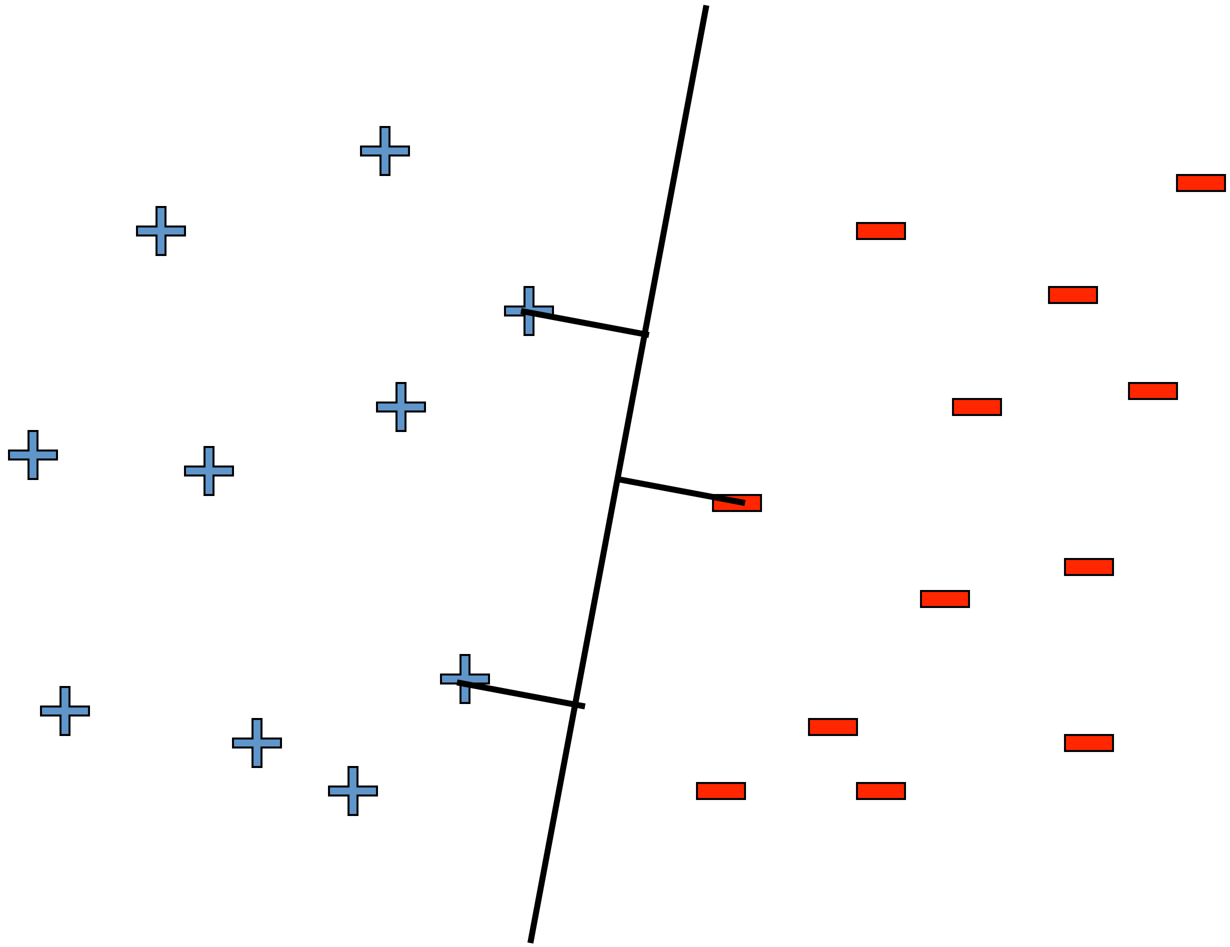
MACHINE LEARNING DEPARTMENT
Carnegie Mellon.
School of Computer Science

# At Pittsburgh G-20 summit …

# Linear classifiers – which line is better?



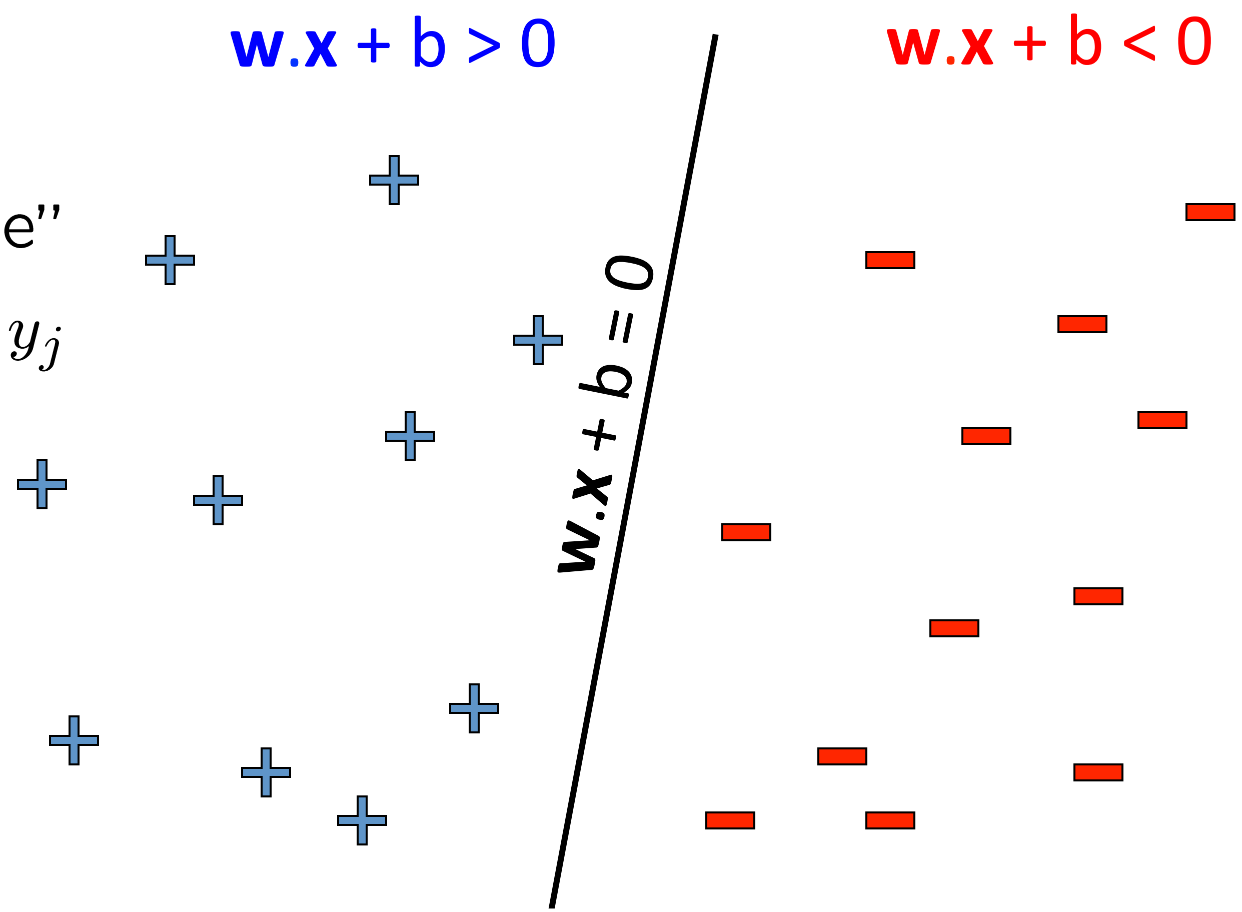
# Pick the one with the largest margin!



# Parameterizing the decision boundary

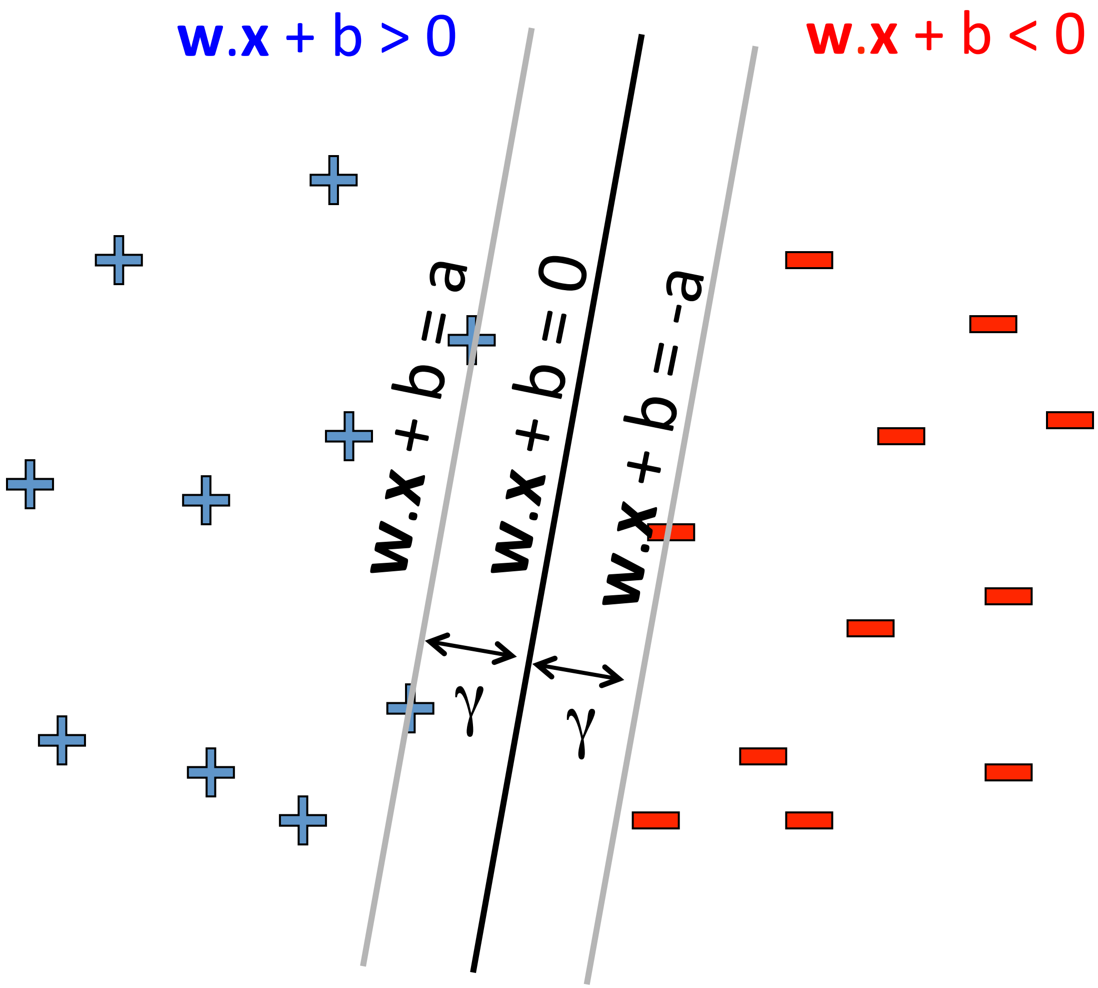
“confidence”

$$\left(\mathbf{w}.\mathbf{x}_j + b\right) y_j$$




**Labels:** $y_i \in \{-1, +1\}$ — class

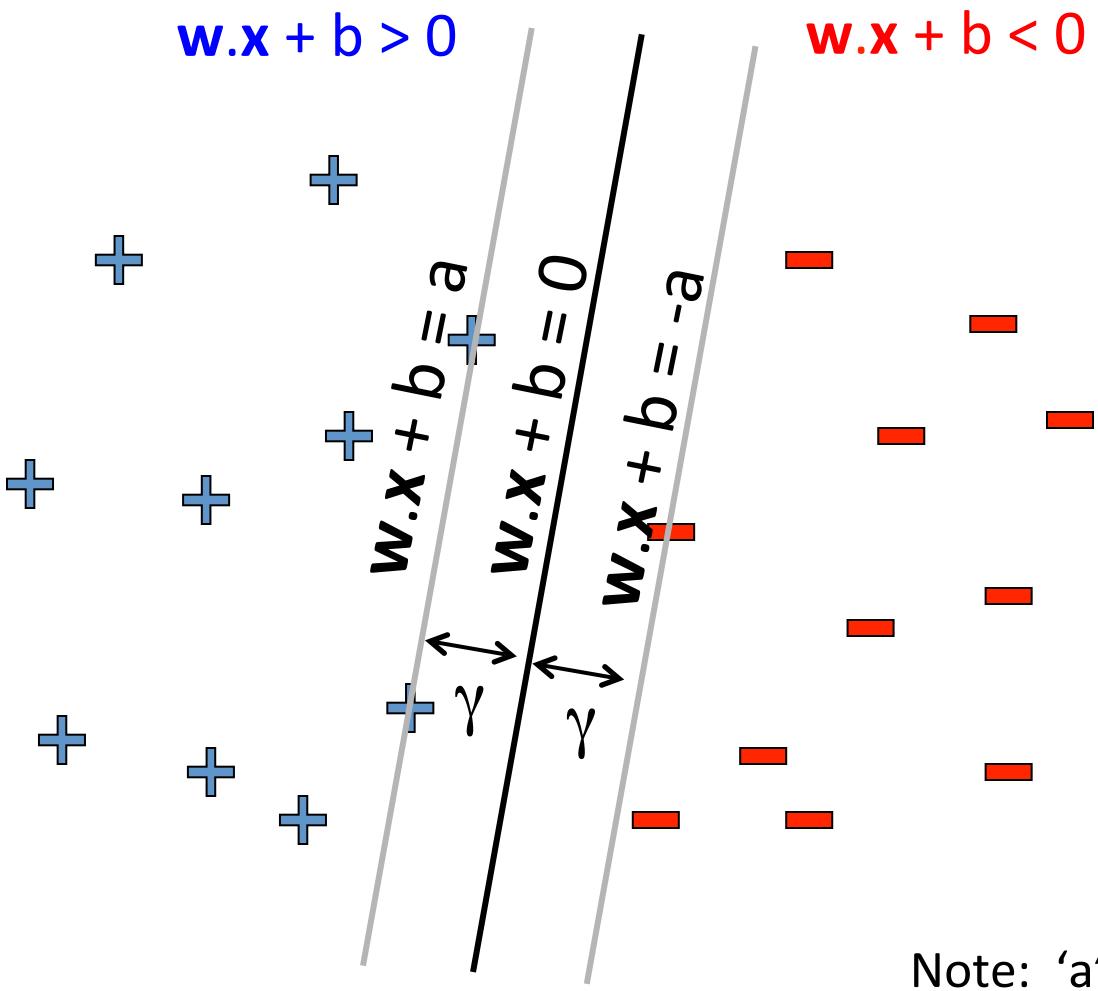
# Maximizing the margin




Distance of closest examples from the line/hyperplane

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

# Maximizing the margin




Distance of closest examples from the line/hyperplane

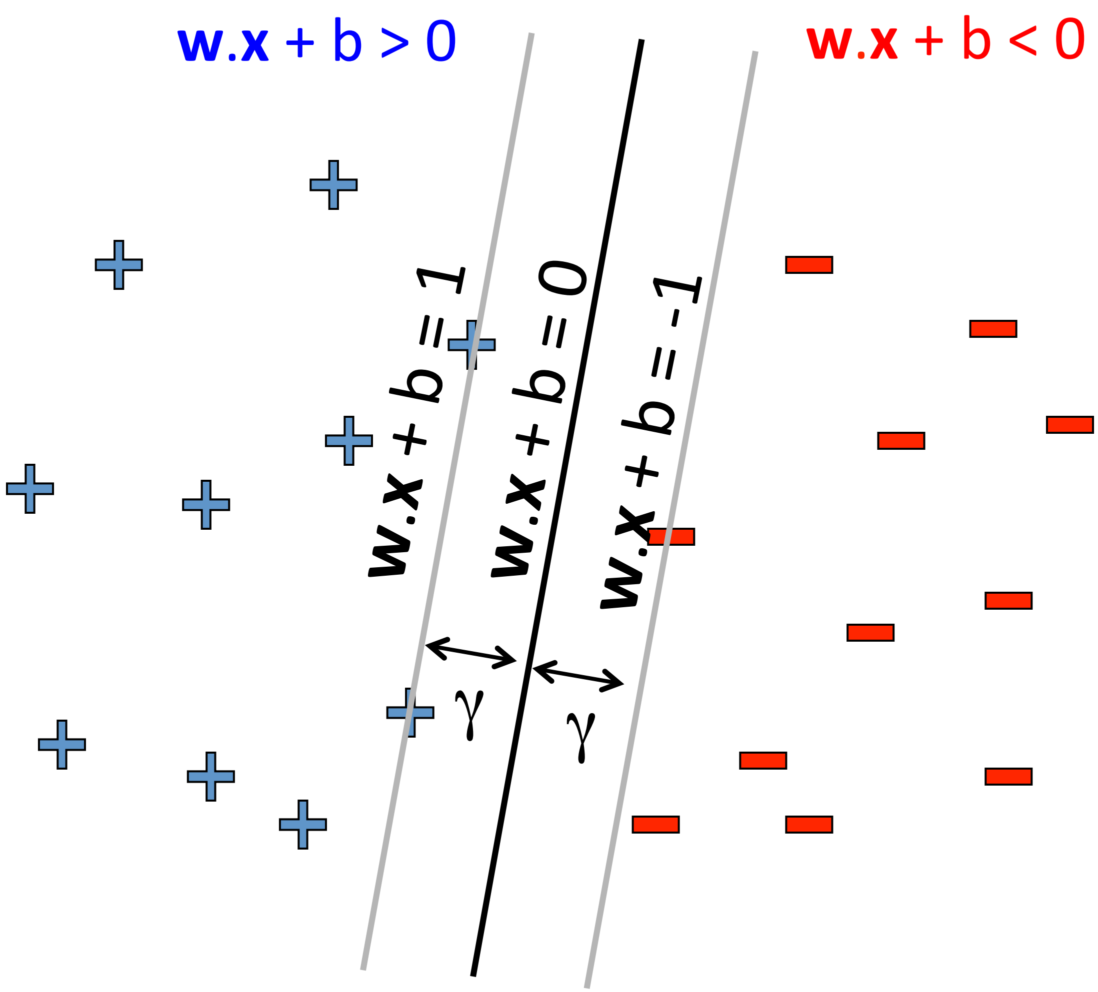
$$\text{margin} = \gamma = a/\|w\|$$

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

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

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

# Support Vector Machines




$$\max_{\mathbf{w},b} \ \gamma = 1/\|w\|$$

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

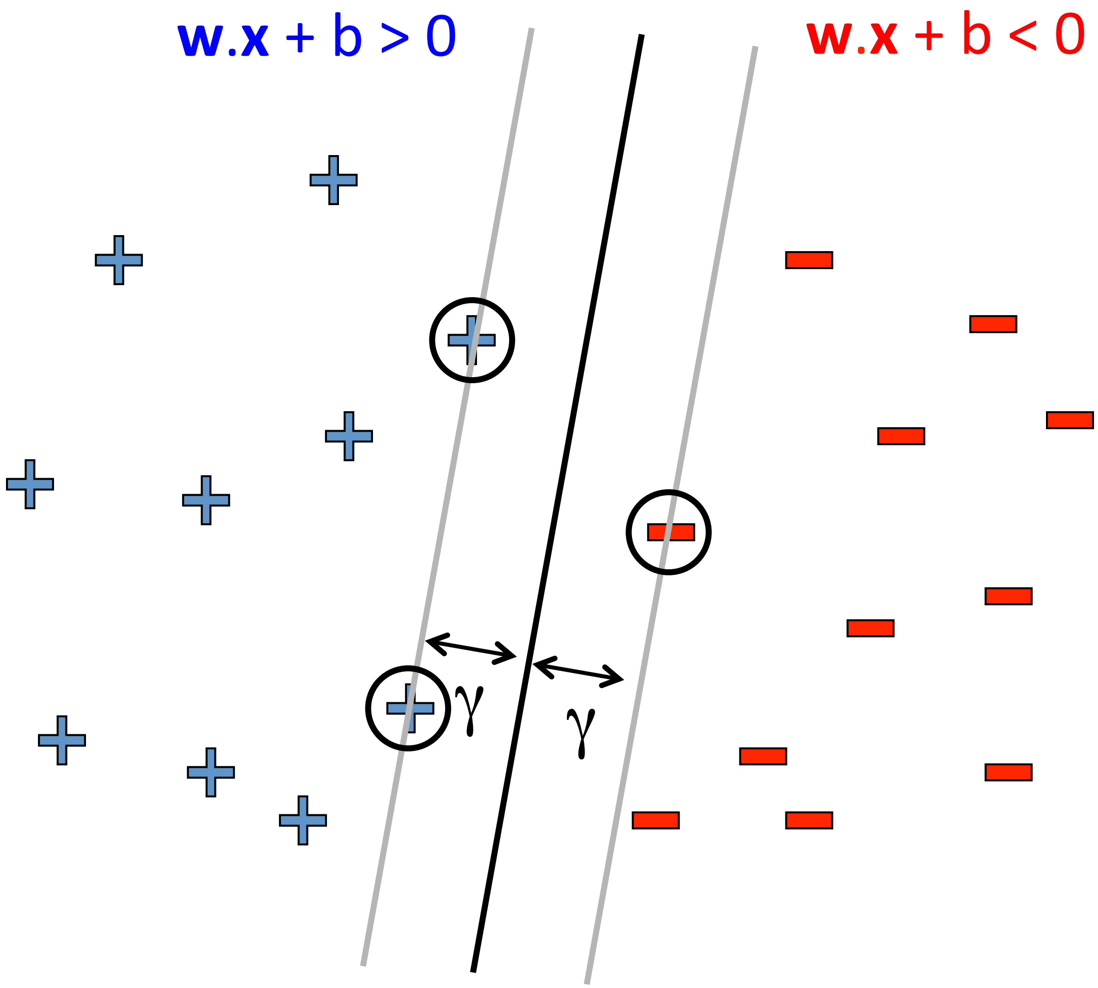
$$\min_{\mathbf{w},b} \ \mathbf{w}.\mathbf{w}$$

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

Solve efficiently by quadratic programming (QP)

- Well-studied solution algorithms

# Support Vectors



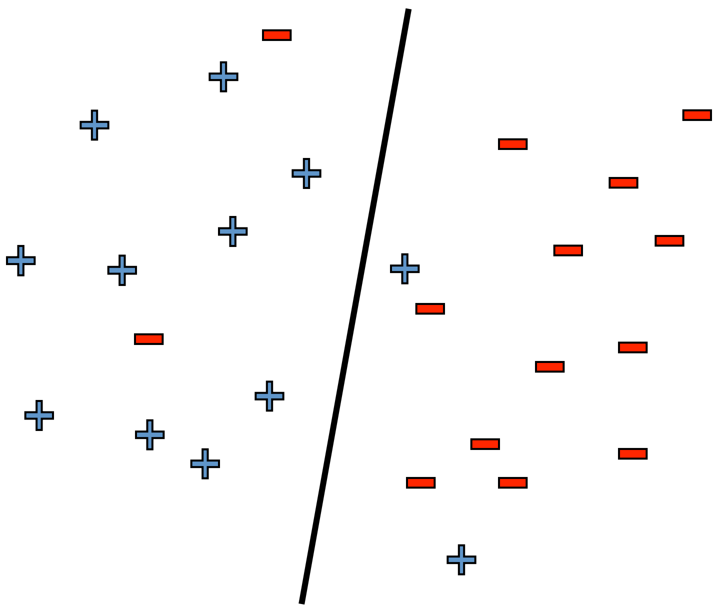

Linear hyperplane defined by "support vectors"

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

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

How many support vectors in linearly separable case, given d dimensions?

# What if data is not linearly separable?



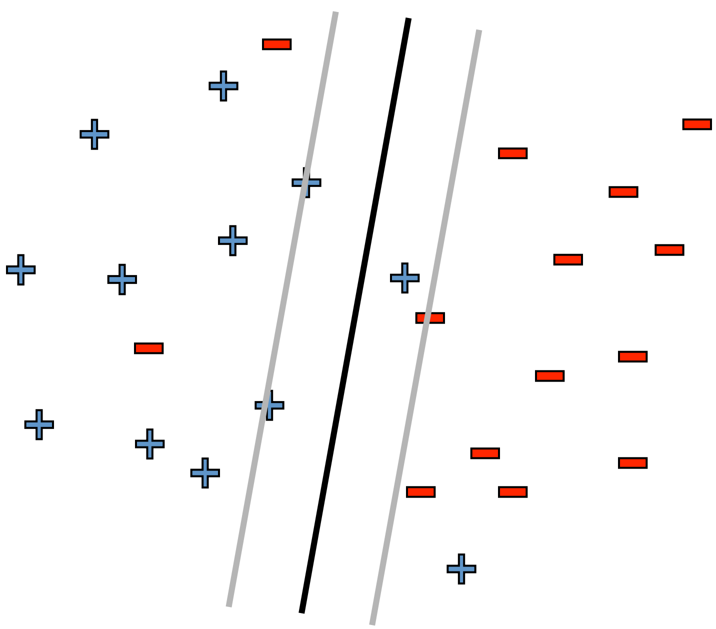
Use features of features of features of features....

$x_1^2, x_2^2, x_1x_2, ...., \exp(x_1)$

But run risk of overfitting!

# What if data is 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

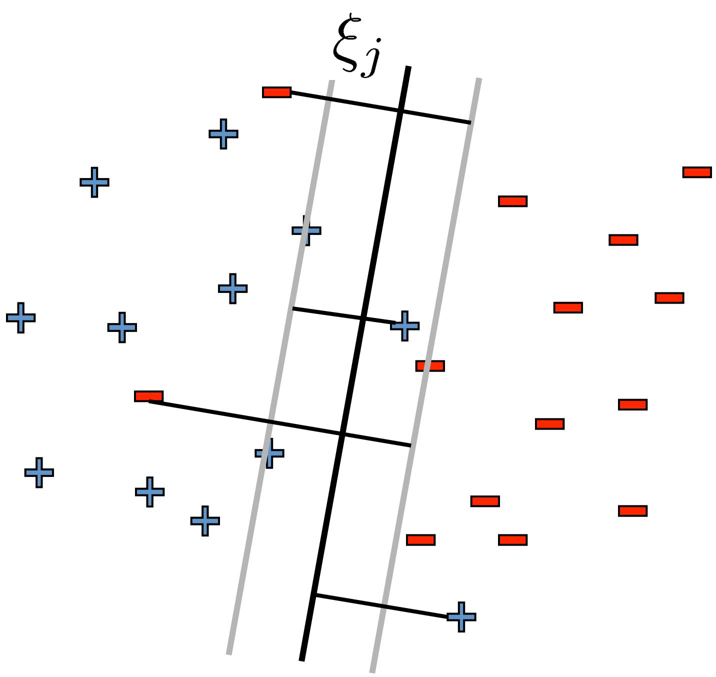
C - tradeoff parameter

Not QP ☹

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

# What if data is not linearly separable?

Allow "error" in classification




**Soft margin approach**

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

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

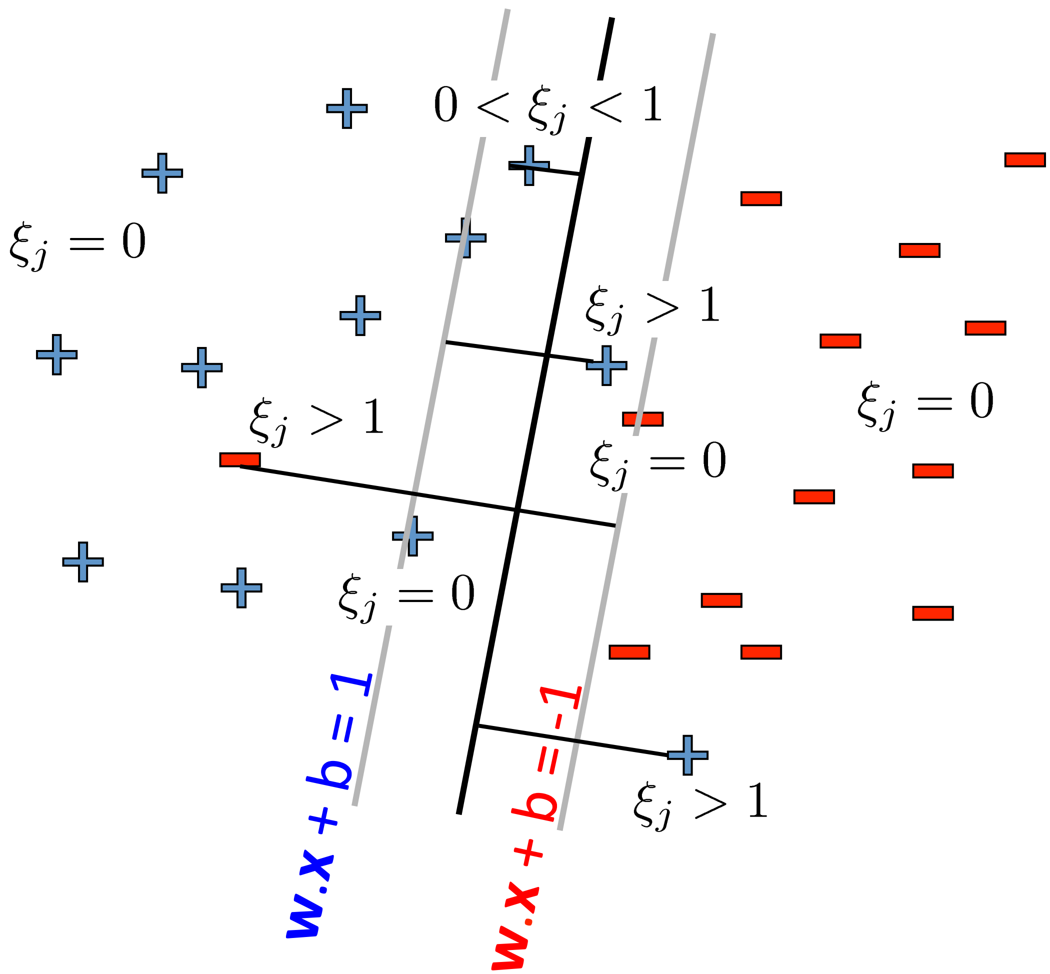
$$\xi_j \geq 0 \quad \forall j$$

$\xi_j$ - "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:

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

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

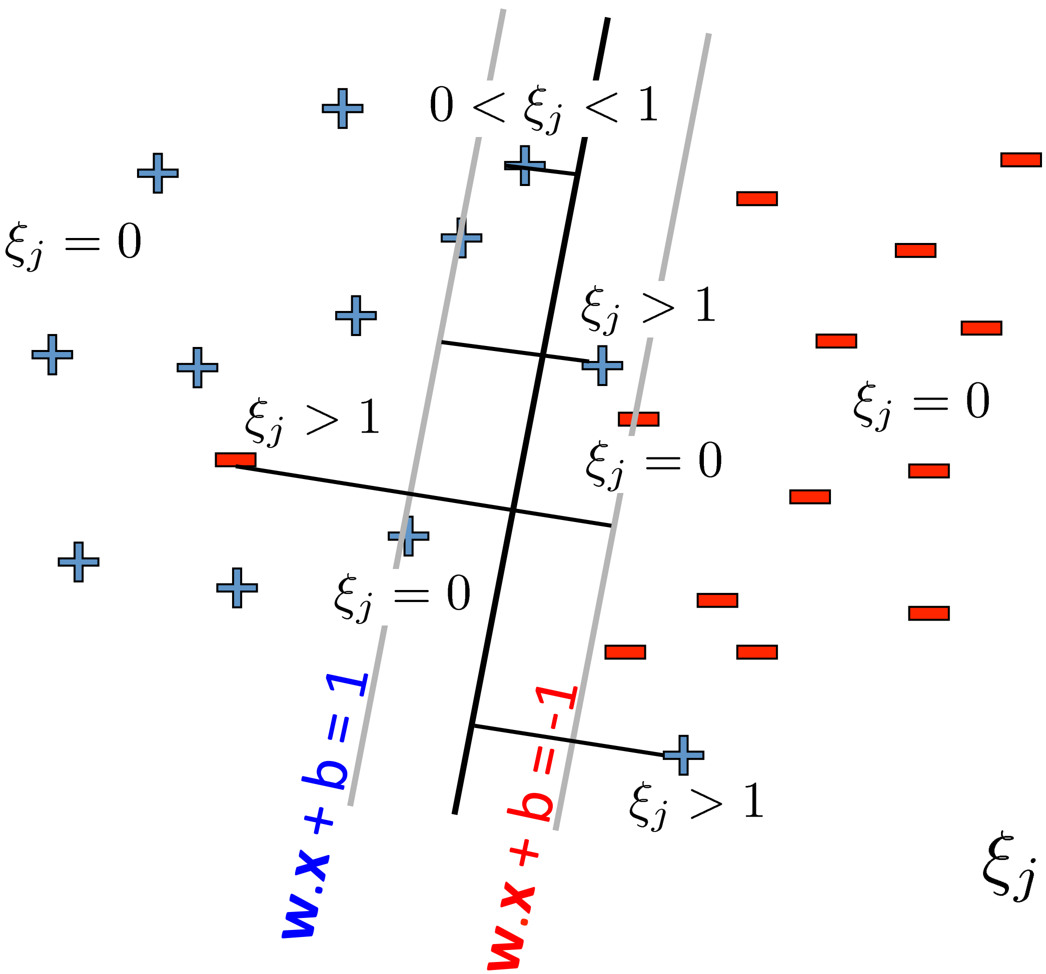
Penalty for misclassifying:

$$C\ \xi_j$$

How do we recover hard margin SVM?

Set C = ∞

# Soft-margin SVM




Soften the constraints:

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

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

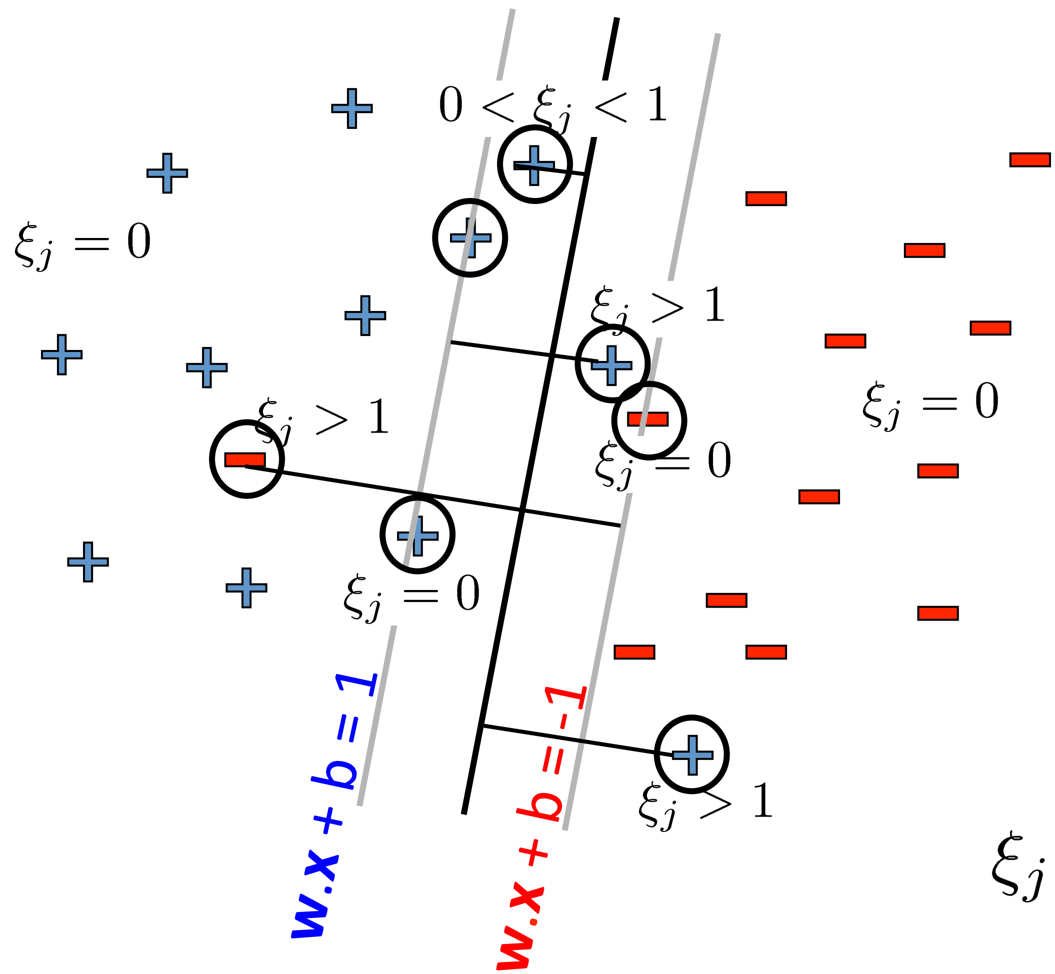
Penalty for misclassifying:

$$C\, \xi_j$$

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

$$a_+ = \max(a, 0)$$

# 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:

$$C\ \xi_j$$

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

$$a_+ = \max(a, 0)$$

# Slack variables – Hinge loss

Regularized loss function

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

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

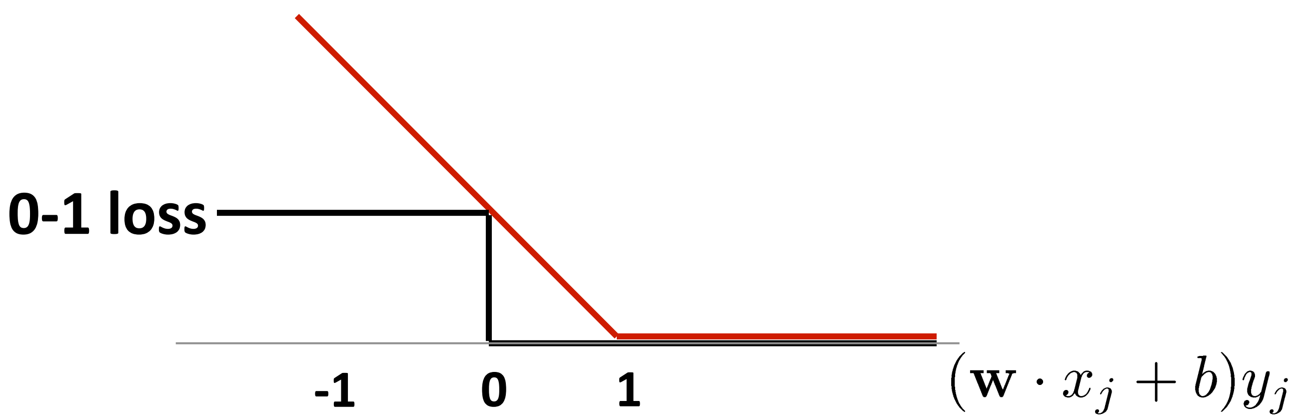
$$\min_{\mathbf{w},b,\xi} \underbrace{\mathbf{w}.\mathbf{w}}_{\text{Regularization}} + C \underbrace{\sum_j \xi_j}_{\text{loss}}$$

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

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

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

**Hinge loss**



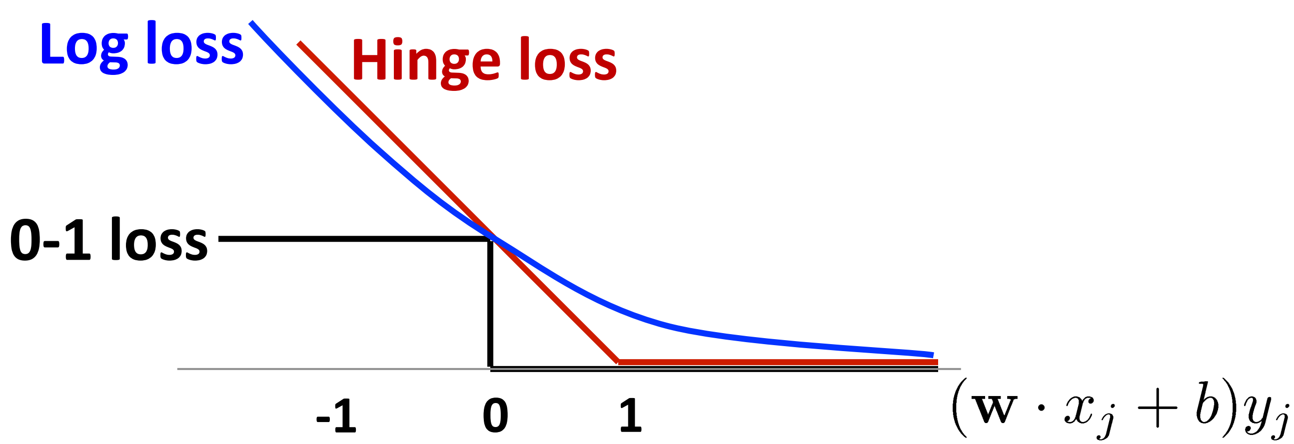
# SVM vs. Logistic Regression

SVM : **Hinge loss**

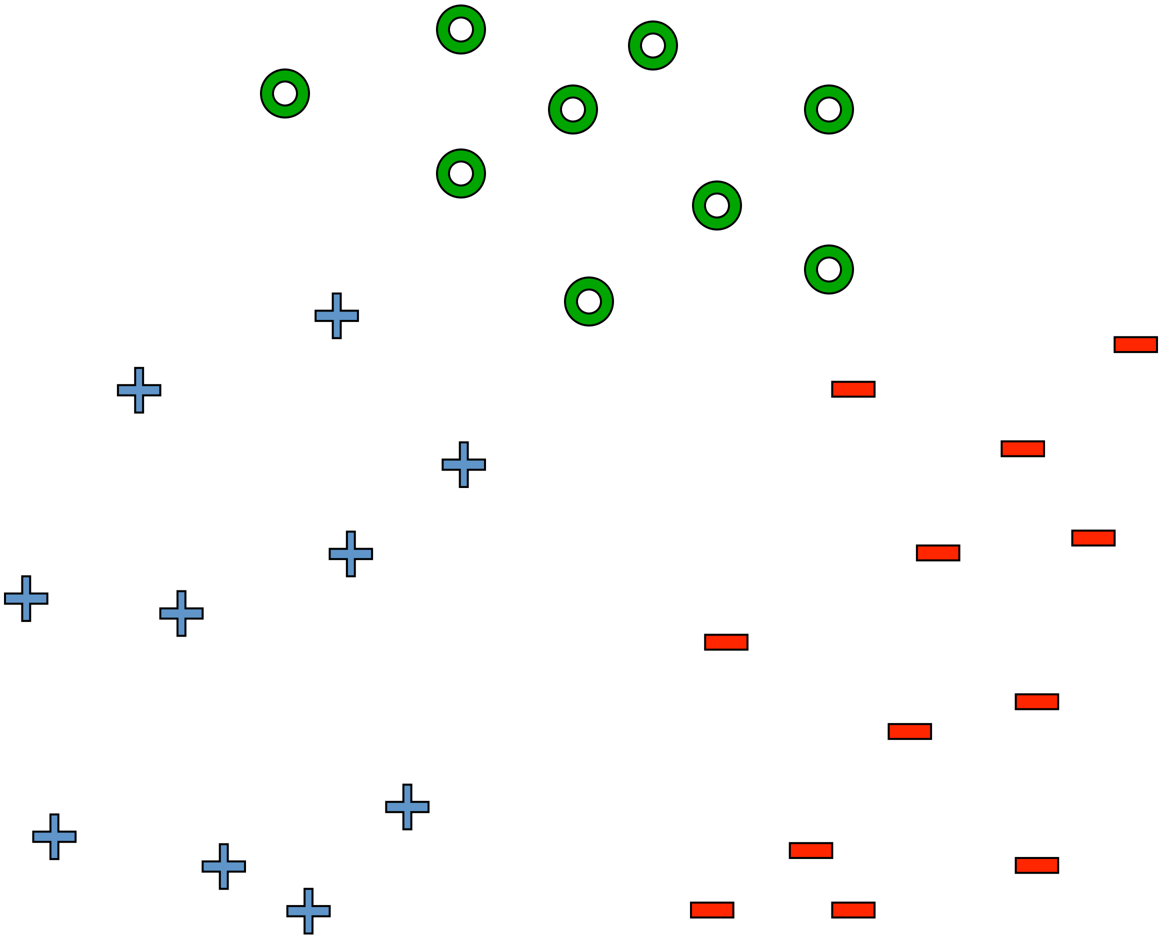
$$\text{loss}(f(x_j), y_j) = (1 - (\mathbf{w} \cdot x_j + b)y_j))_+$$

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

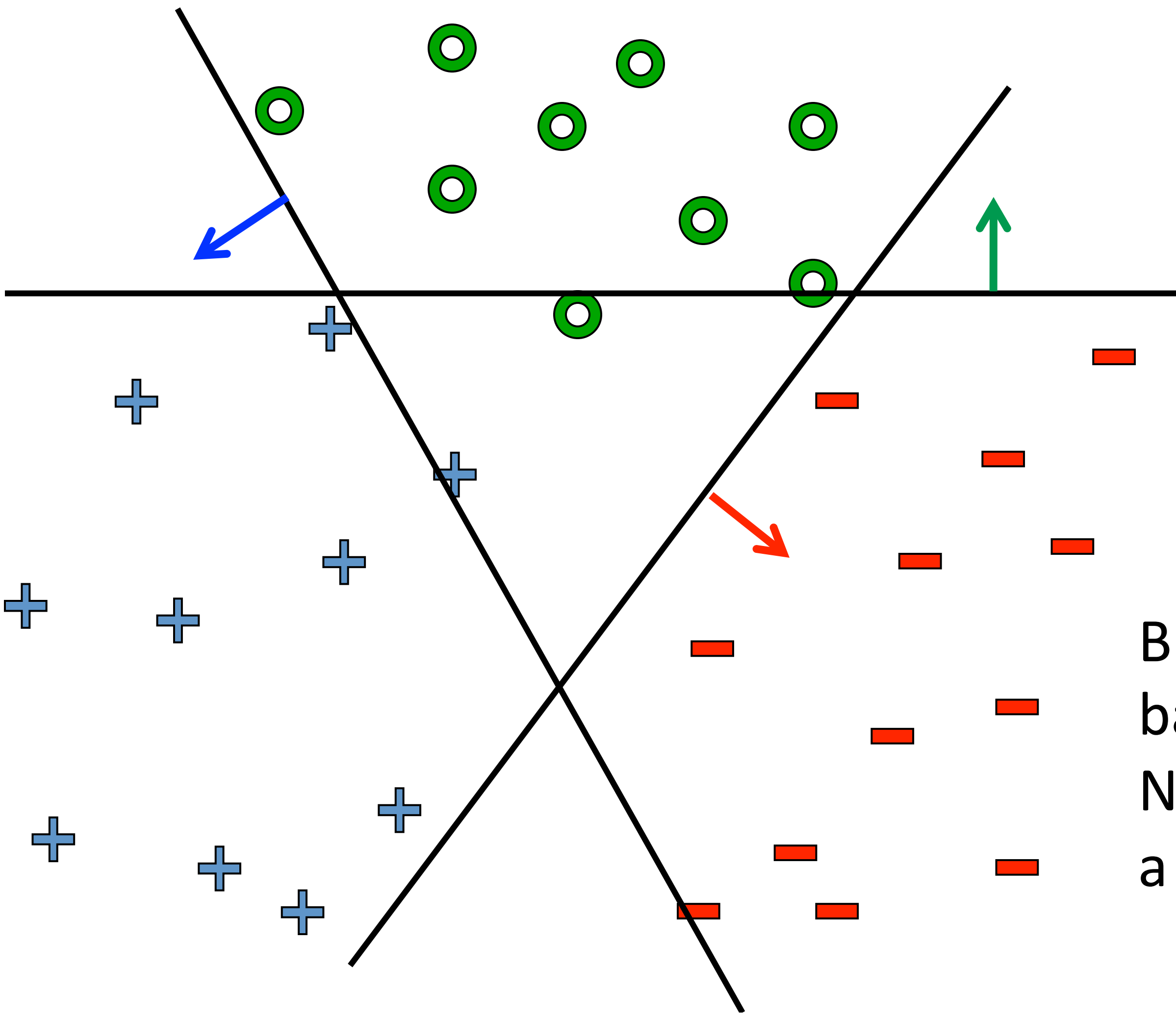
$$\text{loss}(f(x_j), y_j) = -\log P(y_j \mid x_j, \mathbf{w}, b) = \log(1 + e^{-(\mathbf{w}\cdot x_j + b)y_j})$$



# What about multiple classes?



# One against rest



**Learn 3 classifiers separately:**
Class k vs. rest

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

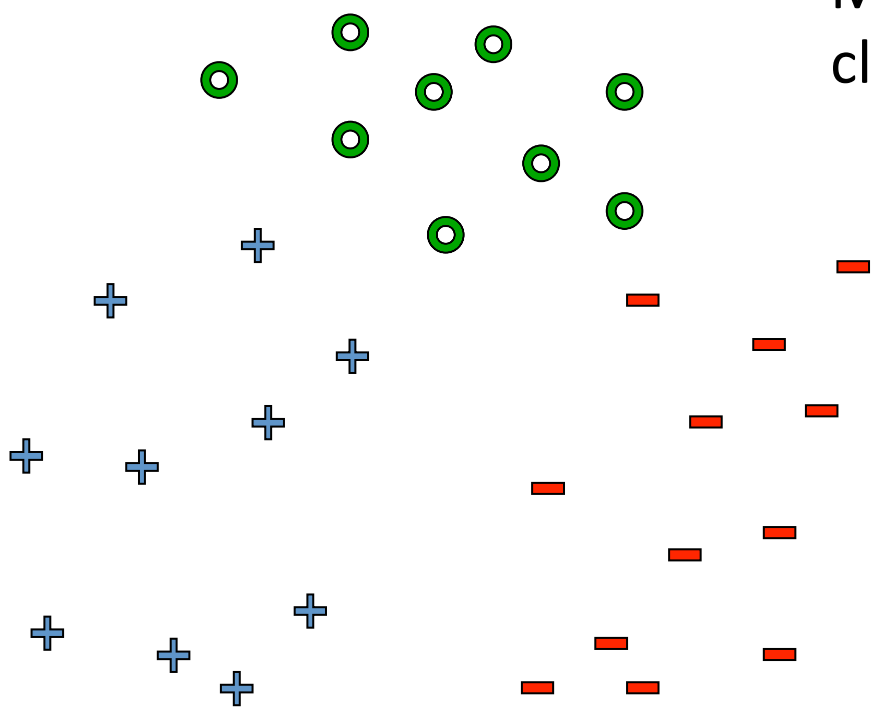
$$y = \arg\max_k \mathbf{w}_k.x + b_k$$

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**

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



Margin - gap between correct class and nearest other class

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

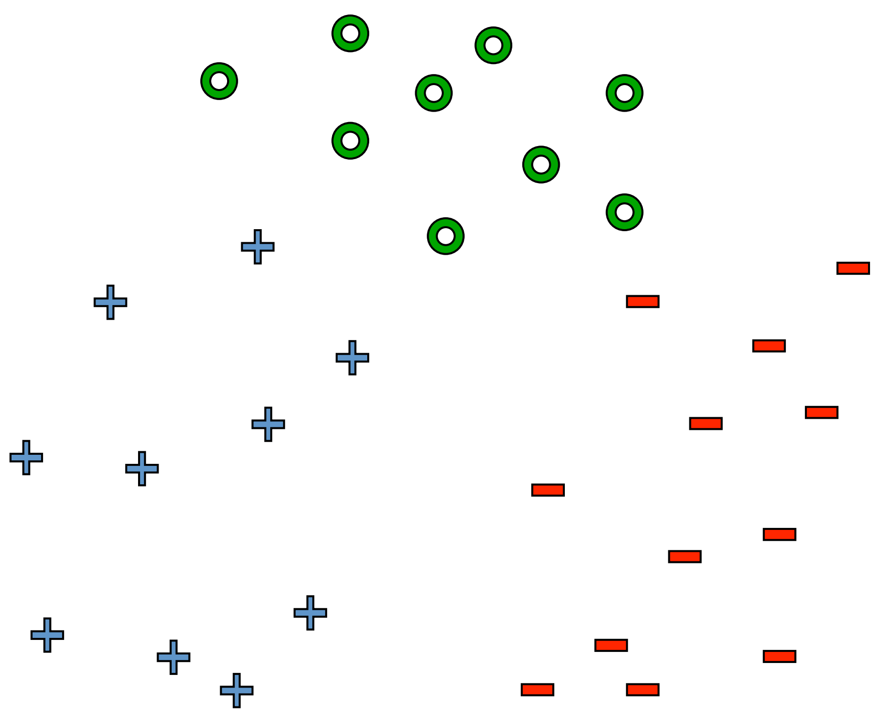
# Learn 1 classifier: Multi-class SVM

**Simultaneously learn 3 sets of weights**

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

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

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



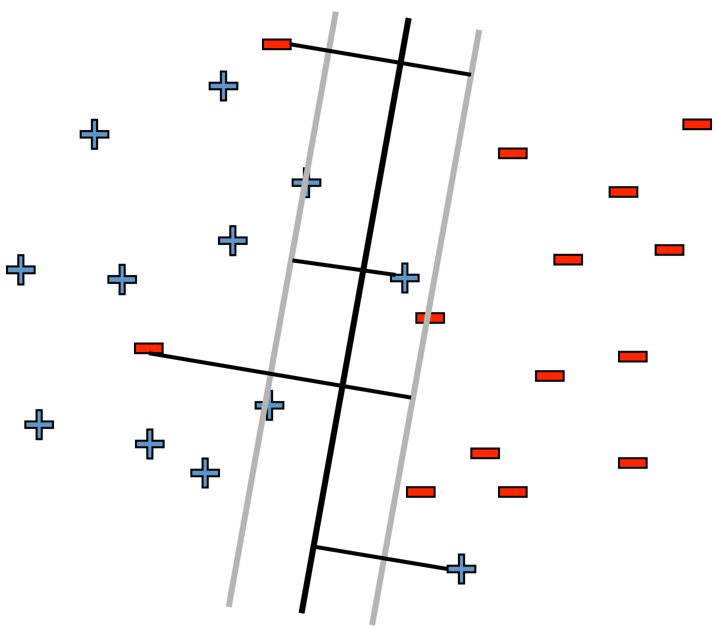
y = arg max $\mathbf{w}^{(k)}$.x + $b^{(k)}$

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

# SVMs reminder



Soft margin approach

$$\min_{\mathbf{w},b,\xi} \; \overbrace{\mathbf{w}.\mathbf{w}}^{\text{Regularization}} + \overbrace{C\,\Sigma\,\xi_j}^{\text{Hinge loss}}$$

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

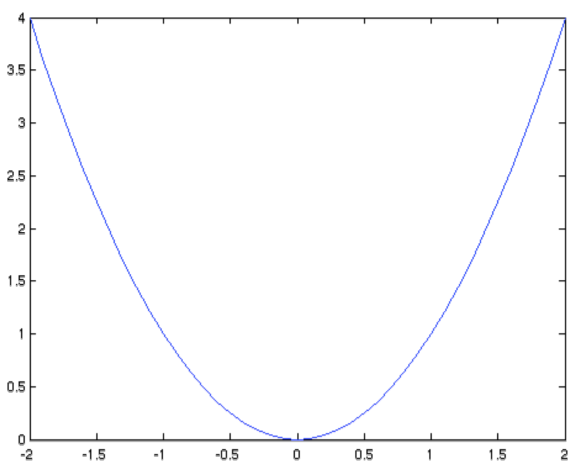
$$\xi_j \geq 0 \quad \forall j$$

Essentially a constrained optimization problem!

# Constrained Optimization

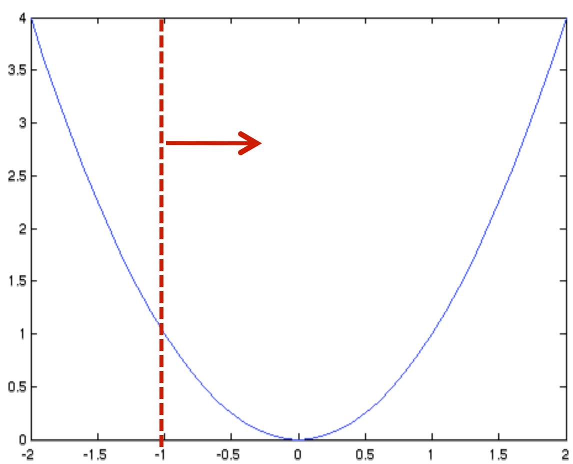
$$\min_x \; x^2$$
$$\text{s.t.} \quad x \geq b$$

$$\min_x \; x^2$$




$$x^* = 0$$

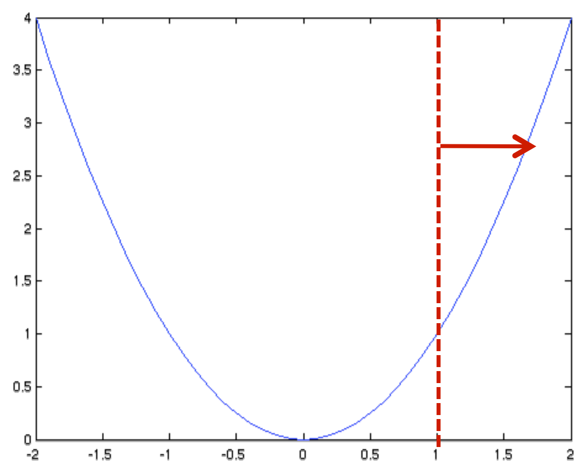
$$\min_x \; x^2$$
$$\text{s.t.} \quad x \geq -1$$




$$x^* = 0$$

Constraint is ineffective

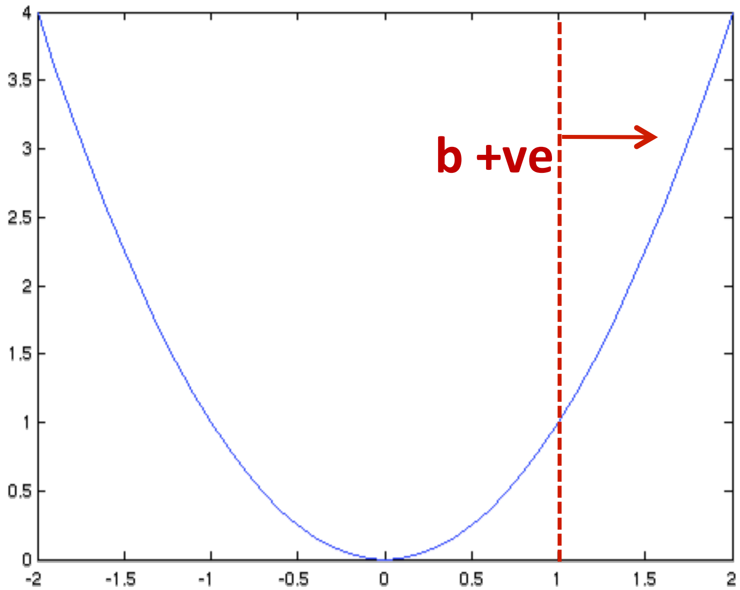
$$\min_x \; x^2$$
$$\text{s.t.} \quad x \geq 1$$




$$x^* = 1$$

Constraint is effective

# Constrained Optimization




$$x^* = b$$

**Primal problem:**

$$\min_x \; x^2 \quad \text{s.t.} \quad x \geq b$$

**Moving the constraint to objective function**

**Lagrangian:**

$$L(x,\alpha) = x^2 - \alpha(x - b) \quad \text{s.t.} \quad \alpha \geq 0$$

**Dual problem:**

$$\max_\alpha \; d(\alpha) \quad \text{s.t.} \quad \alpha \geq 0$$

$$d(\alpha) \rightarrow \min_x L(x,\alpha)$$

**$\alpha$ = 0 constraint is ineffective**

**$\alpha$ > 0 constraint is effective**

# Dual SVM – linearly separable case

- Primal problem: $\text{minimize}_{\mathbf{w},b} \quad \frac{1}{2}\mathbf{w}.\mathbf{w}$

$$\left(\mathbf{w}.\mathbf{x}_j + b\right) y_j \geq 1, \ \ \forall j$$

**w – weights on features**

- Lagrangian:

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

$$\alpha_j \geq 0, \ \ \forall j$$

**$\alpha$ – weights on training pts**

**$\alpha_j = 0$ constraint is ineffective** $(w.x_j+b)y_j > 1$ (not a support vector)

**$\alpha_j > 0$ constraint is effective** $(w.x_j+b)y_j = 1$ (point j is a support vector)

# Dual SVM – linearly separable case

- Dual problem:

$$\max_\alpha \min_{\mathbf{w},b} L(\mathbf{w},b,\alpha) = \frac{1}{2}\mathbf{w}.\mathbf{w} - \sum_j \alpha_j \left[\left(\mathbf{w}.\mathbf{x}_j + b\right) y_j - 1\right]$$

$$\alpha_j \geq 0, \ \ \forall j$$

$$\frac{\partial L}{\partial \mathbf{w}} = 0 \qquad \Rightarrow \mathbf{w} = \sum_j \alpha_j y_j \mathbf{x}_j$$

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

If we can solve for $\alpha$s (dual problem), then we have a solution for **w**,b (primal problem)

# Dual SVM – linearly separable case

- Dual problem:

$$\max_\alpha \min_{\mathbf{w},b} L(\mathbf{w},b,\alpha) = \frac{1}{2}\mathbf{w}.\mathbf{w} - \sum_j \alpha_j \left[\left(\mathbf{w}.\mathbf{x}_j + b\right) y_j - 1\right]$$

$$\alpha_j \geq 0, \;\; \forall j$$

$$\mathbf{w} = \sum_j \alpha_j y_j \mathbf{x}_j \qquad \sum_j \alpha_j y_j = 0$$

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

$$\sum_i \alpha_i y_i = 0$$

$$\alpha_i \geq 0$$

# Dual SVM – linearly separable case

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

$$\sum_i \alpha_i y_i = 0$$
$$\alpha_i \geq 0$$

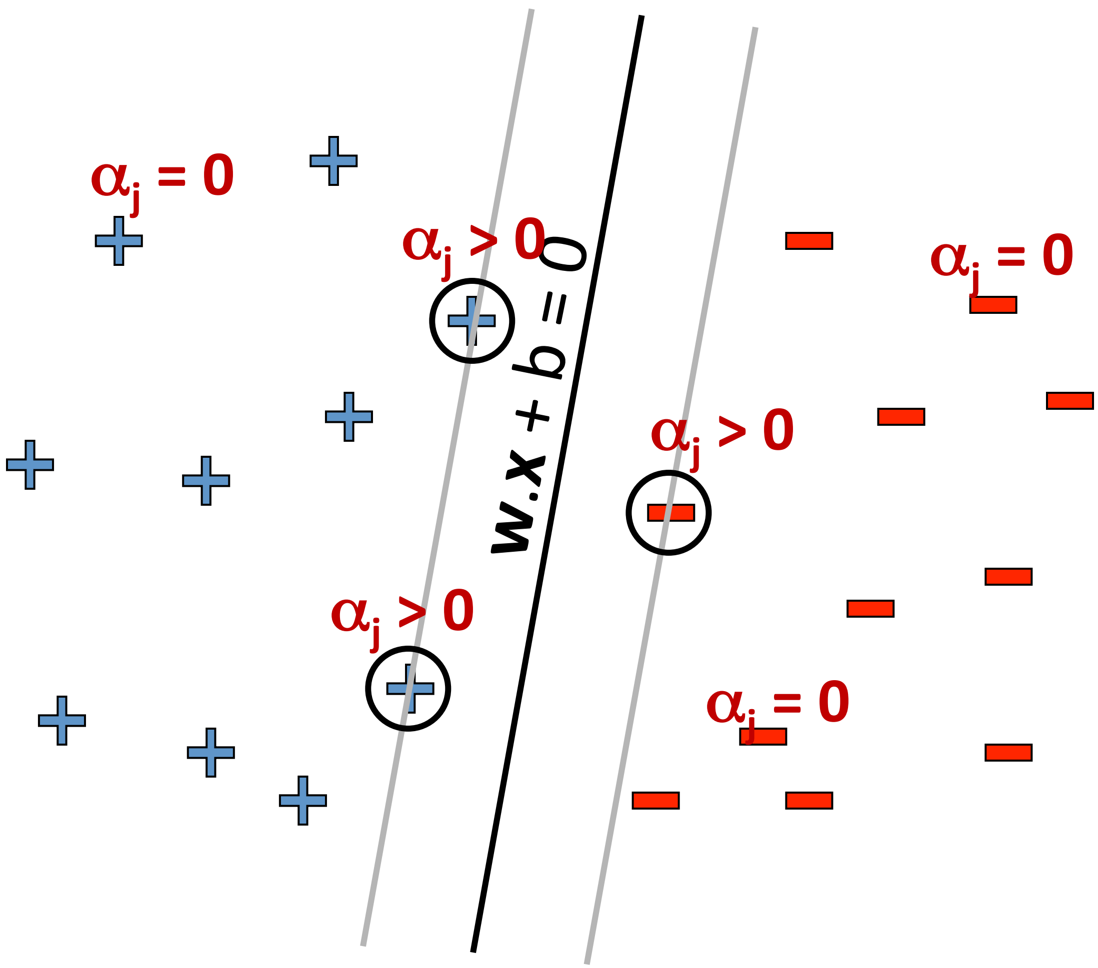
Dual problem is also QP

Solution gives $\alpha_j$s $\longrightarrow$

$$\boxed{\begin{array}{l} \mathbf{w} = \sum_i \alpha_i y_i \mathbf{x}_i \\ b = y_k - \mathbf{w}.\mathbf{x}_k \\ \text{for any } k \text{ where } \alpha_k > 0 \end{array}}$$

$w.x_k + b = y_k$ ⇦ $(w.x_k + b)y_k = 1$ ⇦ Use support vectors to compute b

# Dual SVM Interpretation: Sparsity




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

Only few $\alpha_j$s can be non-zero : where constraint is tight

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

**Support vectors** – training points j whose $\alpha_j$s are non-zero

# Dual SVM – non-separable case

- Primal problem:

$$\text{minimize}_{\mathbf{w},b} \quad \frac{1}{2}\mathbf{w}.\mathbf{w} + C\sum_j \xi_j$$
$$\left(\mathbf{w}.\mathbf{x}_j + b\right) y_j \geq 1 - \xi_j, \ \forall j$$
$$\xi_j \geq 0, \ \forall j$$

$$\boxed{\begin{matrix}\alpha_j \\ \mu_j\end{matrix}}$$

**Lagrange Multipliers**

- Dual problem:

$$\max_{\alpha,\mu} \min_{\mathbf{w},b} L(\mathbf{w}, b, \alpha, \mu)$$
$$s.t. \alpha_j \geq 0 \quad \forall j$$
$$\mu_j \geq 0 \quad \forall j$$

# Dual SVM – non-separable case

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

$\sum_i \alpha_i y_i = 0$

$\boxed{C \geq} \alpha_i \geq 0$

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

Intuition:

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

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

Dual problem is also QP

Solution gives $\alpha_j$s $\longrightarrow$

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

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

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

# So why solve the dual SVM?

- There are some quadratic programming algorithms that can solve the dual faster than the primal, specially in high dimensions m>>n

- But, more importantly, the "**kernel trick**"!!!