

Lecture 2: The SVM classifier

C19 Machine Learning

Hilary 2015

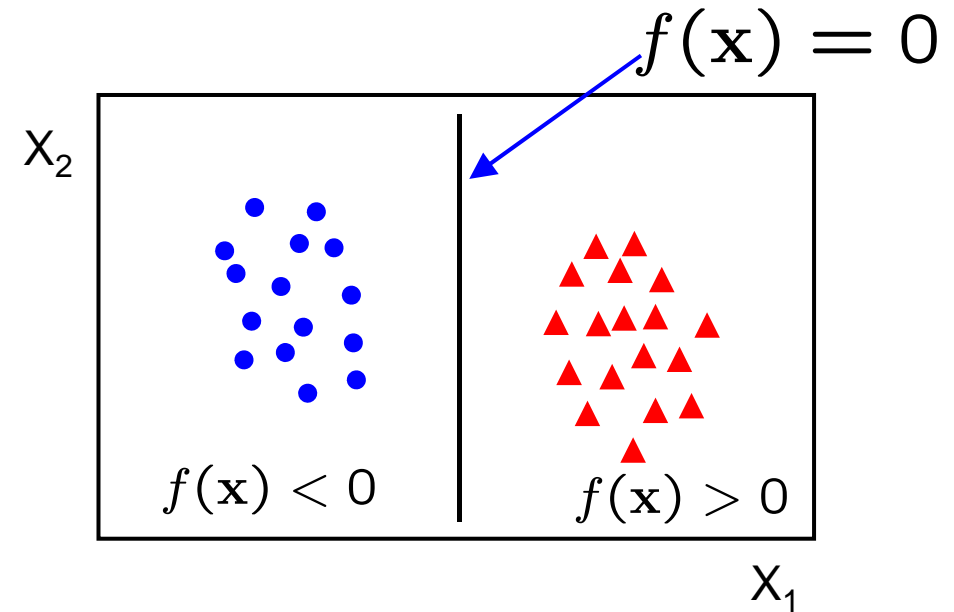
A. Zisserman

- Review of linear classifiers
 - Linear separability
 - Perceptron
- Support Vector Machine (SVM) classifier
 - Wide margin
 - Cost function
 - Slack variables
 - Loss functions revisited
 - Optimization

Linear classifiers

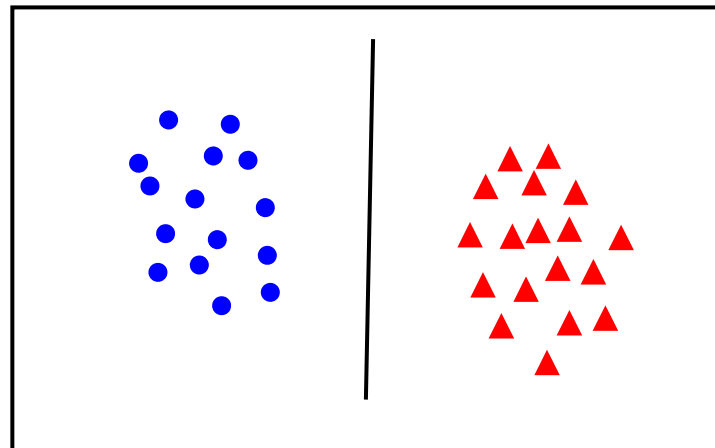
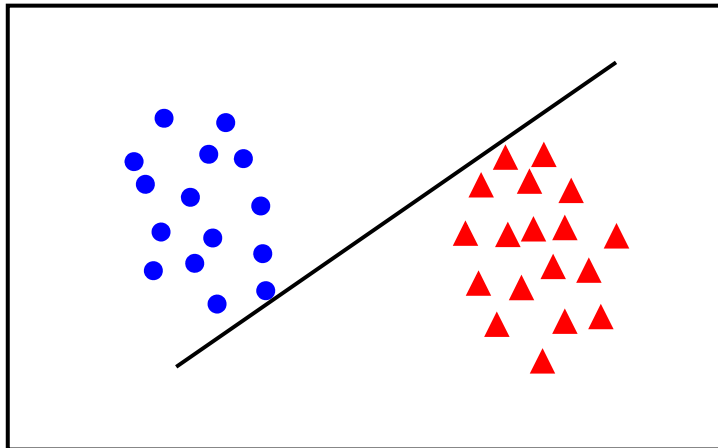
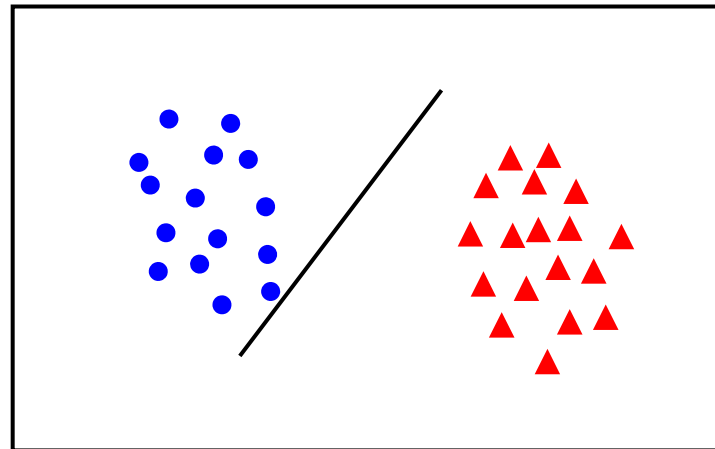
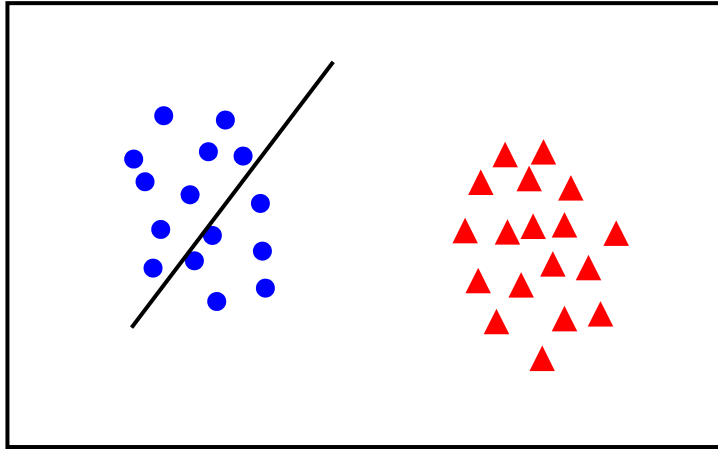
A linear classifier has the form

$$f(\mathbf{x}) = \mathbf{w}^\top \mathbf{x} + b$$



- in 2D the discriminant is a line
- \mathbf{W} is the **normal** to the line, and b the **bias**
- \mathbf{W} is known as the **weight vector**

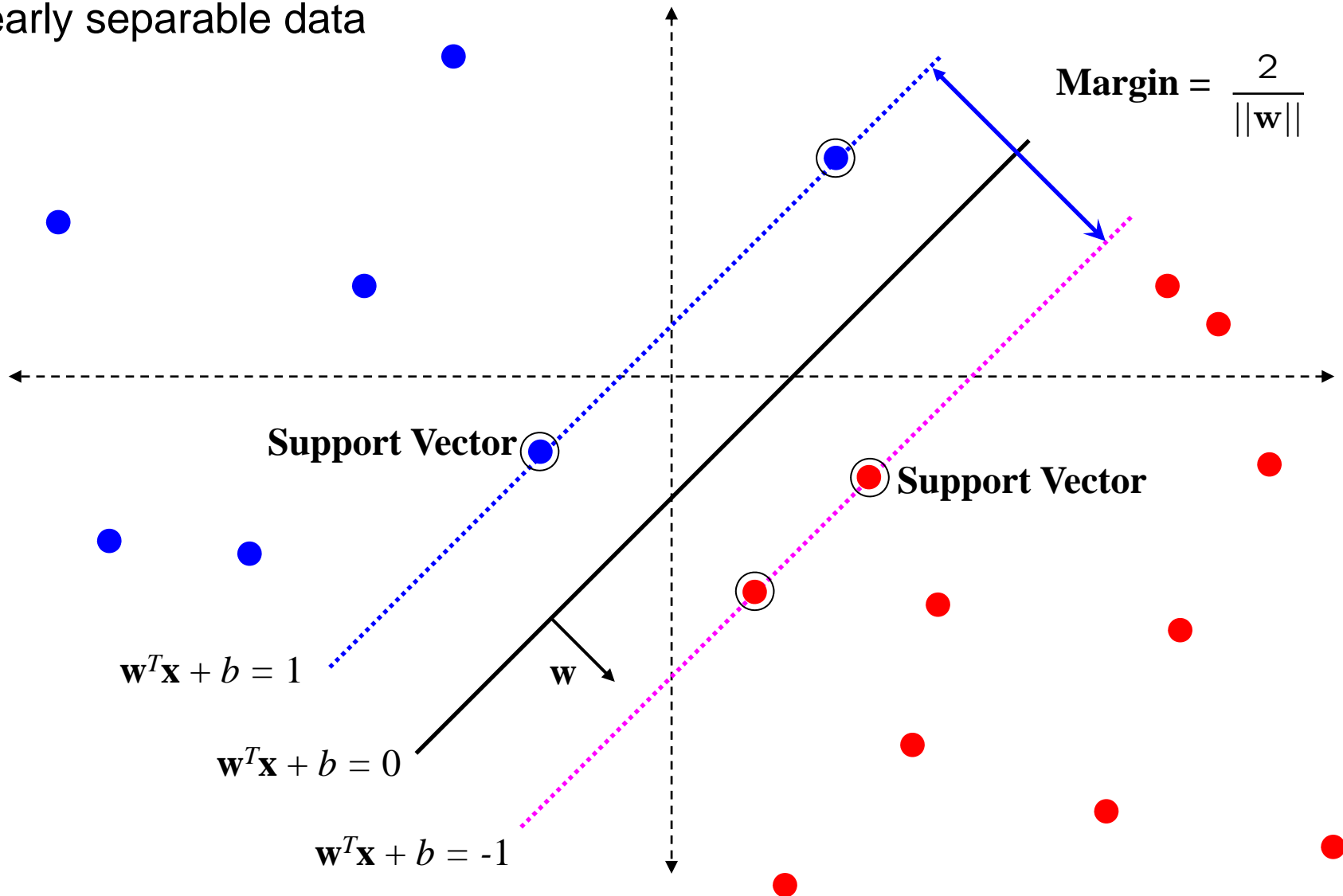
What is the best w ?



- **maximum margin** solution: most stable under perturbations of the inputs

Support Vector Machine

linearly separable data



SVM – Optimization

- Learning the SVM can be formulated as an optimization:

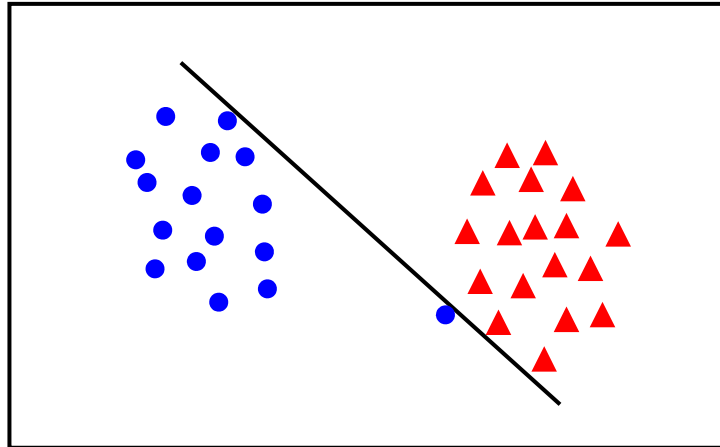
$$\max_{\mathbf{w}} \frac{2}{\|\mathbf{w}\|} \quad \text{subject to } \mathbf{w}^\top \mathbf{x}_i + b \begin{cases} \geq 1 & \text{if } y_i = +1 \\ \leq -1 & \text{if } y_i = -1 \end{cases} \quad \text{for } i = 1 \dots N$$

- Or equivalently

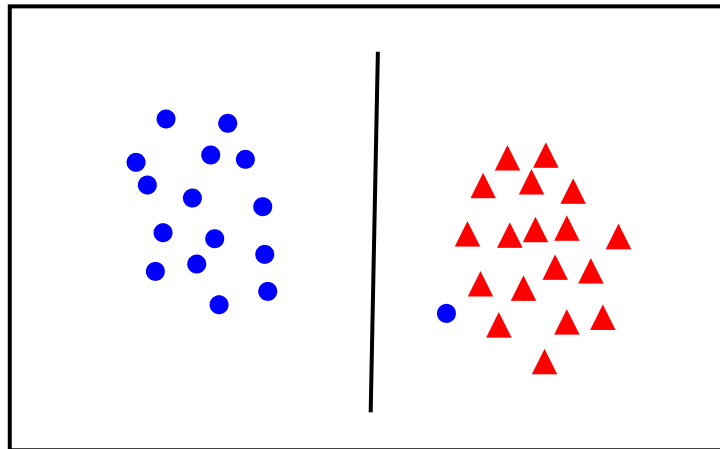
$$\min_{\mathbf{w}} \|\mathbf{w}\|^2 \quad \text{subject to } y_i (\mathbf{w}^\top \mathbf{x}_i + b) \geq 1 \quad \text{for } i = 1 \dots N$$

- This is a quadratic optimization problem subject to linear constraints and there is a unique minimum

Linear separability again: What is the best w ?



- the points can be linearly separated but there is a very narrow margin



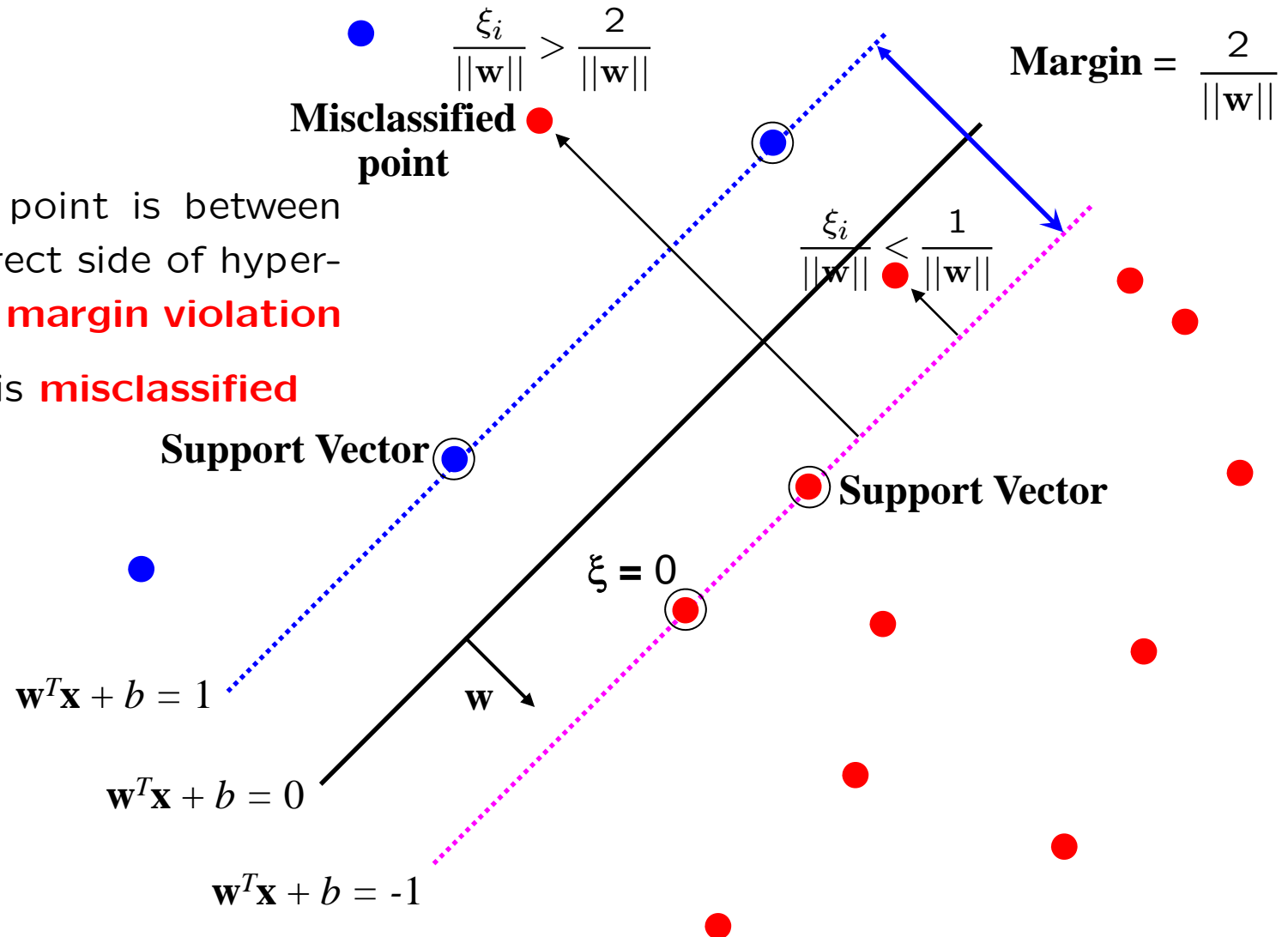
- but possibly the large margin solution is better, even though one constraint is violated

In general there is a trade off between the margin and the number of mistakes on the training data

Introduce “slack” variables

$$\xi_i \geq 0$$

- for $0 < \xi \leq 1$ point is between margin and correct side of hyper-plane. This is a **margin violation**
- for $\xi > 1$ point is **misclassified**



“Soft” margin solution

The optimization problem becomes

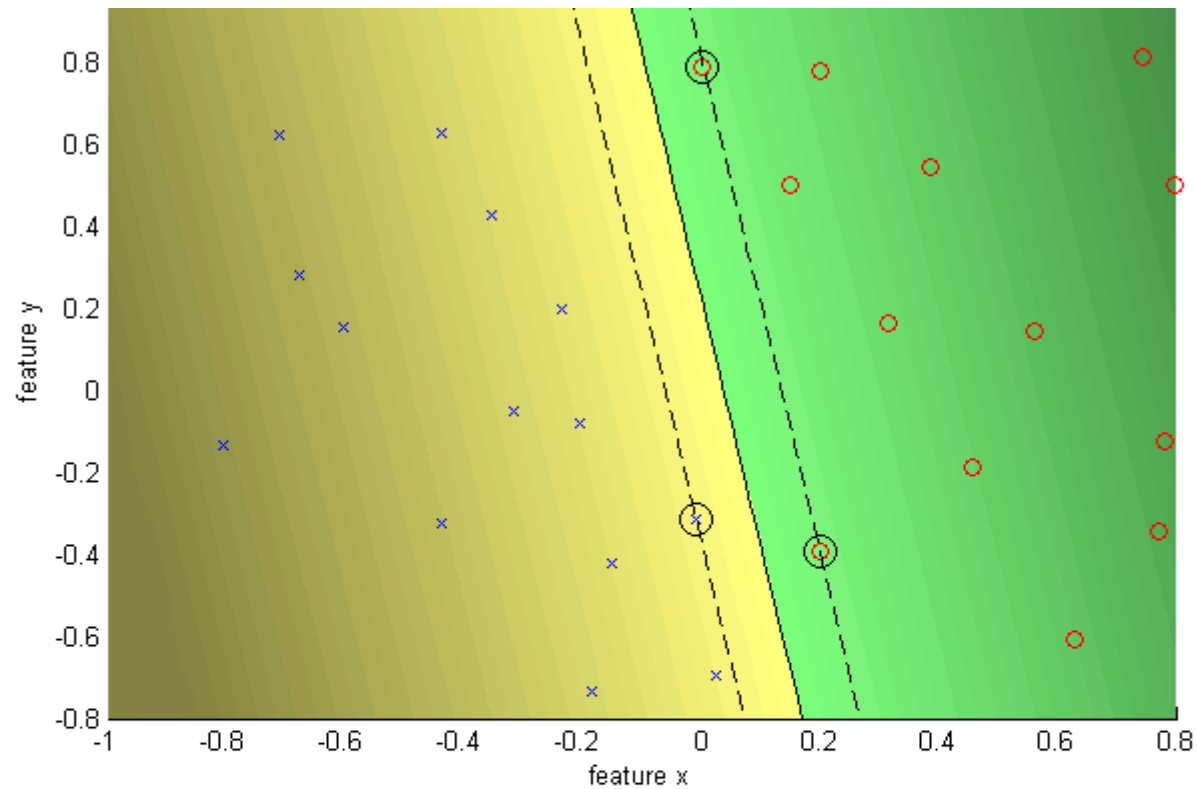
$$\min_{\mathbf{w} \in \mathbb{R}^d, \xi_i \in \mathbb{R}^+} ||\mathbf{w}'||^2 + C \sum_i^N \xi_i$$

subject to

$$y_i (\mathbf{w}' \mathbf{x}_i + b) \geq 1 - \xi_i \text{ for } i = 1 \dots N$$

- Every constraint can be satisfied if ξ_i is sufficiently large
- C is a **regularization** parameter:
 - small C allows constraints to be easily ignored \rightarrow large margin
 - large C makes constraints hard to ignore \rightarrow narrow margin
 - $C = \infty$ enforces all constraints: hard margin
- This is still a quadratic optimization problem and there is a unique minimum. Note, there is only one parameter, C .

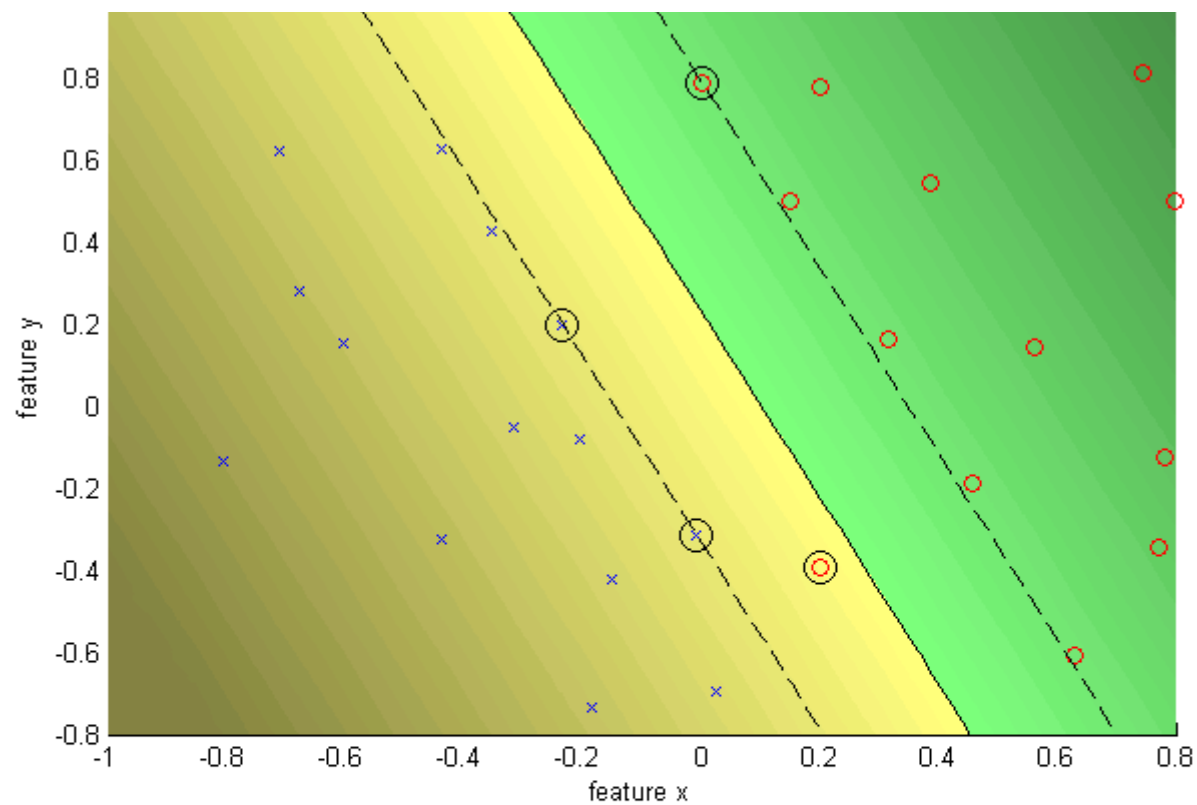
$C = \text{Infinity}$ hard margin



Comment Window

SVM (L1) by Sequential Minimal Optimizer
Kernel: linear (-), C: Inf
Kernel evaluations: 971
Number of Support Vectors: 3
Margin: 0.0966
Training error: 0.00%

$C = 10$ soft margin



Comment Window

SVM (L1) by Sequential Minimal Optimizer
Kernel: linear (-), C: 10.0000
Kernel evaluations: 2645
Number of Support Vectors: 4
Margin: 0.2265
Training error: 3.70%

Optimization

Learning an SVM has been formulated as a **constrained** optimization problem over \mathbf{w} and ξ

$$\min_{\mathbf{w} \in \mathbb{R}^d, \xi_i \in \mathbb{R}^+} \|\mathbf{w}\|^2 + C \sum_i^N \xi_i \text{ subject to } y_i (\mathbf{w}^\top \mathbf{x}_i + b) \geq 1 - \xi_i \text{ for } i = 1 \dots N$$

The constraint $y_i (\mathbf{w}^\top \mathbf{x}_i + b) \geq 1 - \xi_i$, can be written more concisely as

$$y_i f(\mathbf{x}_i) \geq 1 - \xi_i$$

which, together with $\xi_i \geq 0$, is equivalent to

$$\xi_i = \max(0, 1 - y_i f(\mathbf{x}_i))$$

Hence the learning problem is equivalent to the **unconstrained** optimization problem over \mathbf{w}

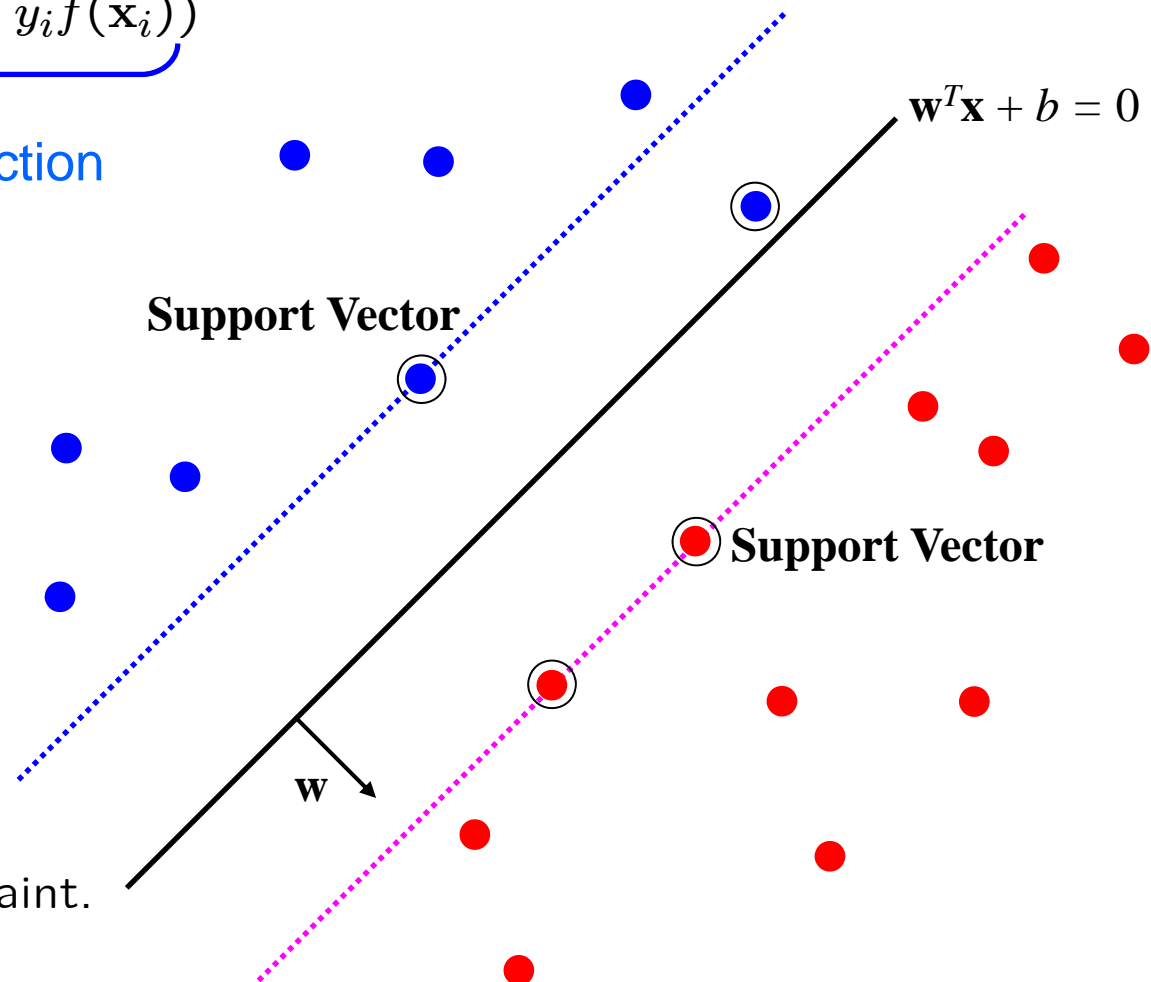
$$\min_{\mathbf{w} \in \mathbb{R}^d} \underbrace{\|\mathbf{w}\|^2}_{\text{regularization}} + C \sum_i^N \underbrace{\max(0, 1 - y_i f(\mathbf{x}_i))}_{\text{loss function}}$$

Loss function

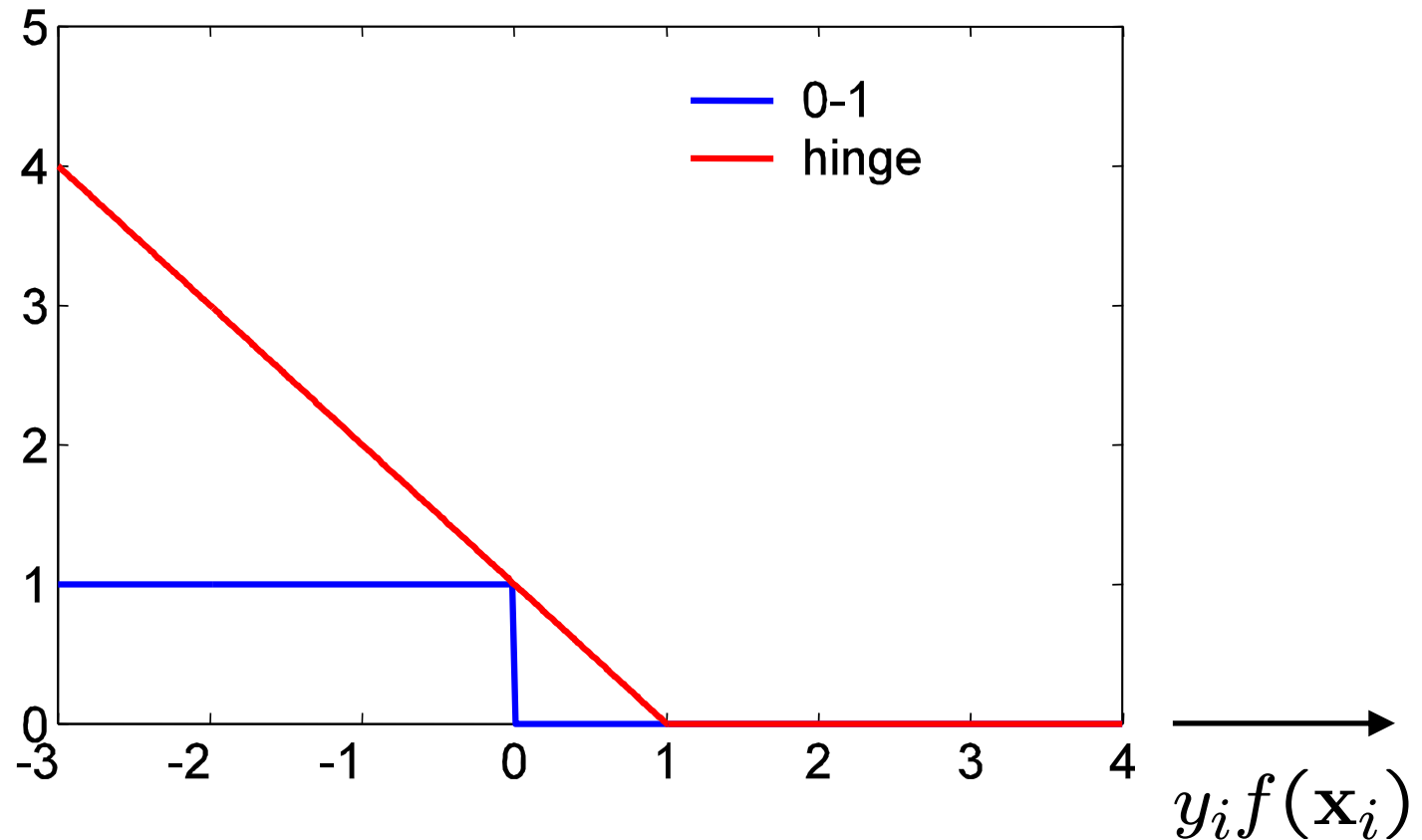
$$\min_{\mathbf{w} \in \mathbb{R}^d} \|\mathbf{w}\|^2 + C \sum_i \underbrace{\max(0, 1 - y_i f(\mathbf{x}_i))}_{\text{loss function}}$$

Points are in three categories:

1. $y_i f(x_i) > 1$
Point is outside margin.
No contribution to loss
2. $y_i f(x_i) = 1$
Point is on margin.
No contribution to loss.
As in hard margin case.
3. $y_i f(x_i) < 1$
Point violates margin constraint.
Contributes to loss



Loss functions



- SVM uses “hinge” loss $\max(0, 1 - y_i f(\mathbf{x}_i))$
- an approximation to the 0-1 loss

Gradient (or steepest) descent algorithm for SVM

To minimize a cost function $\mathcal{C}(\mathbf{w})$ use the iterative update

$$\mathbf{w}_{t+1} \leftarrow \mathbf{w}_t - \eta_t \nabla_{\mathbf{w}} \mathcal{C}(\mathbf{w}_t)$$

where η is the learning rate.

First, rewrite the optimization problem as an **average**

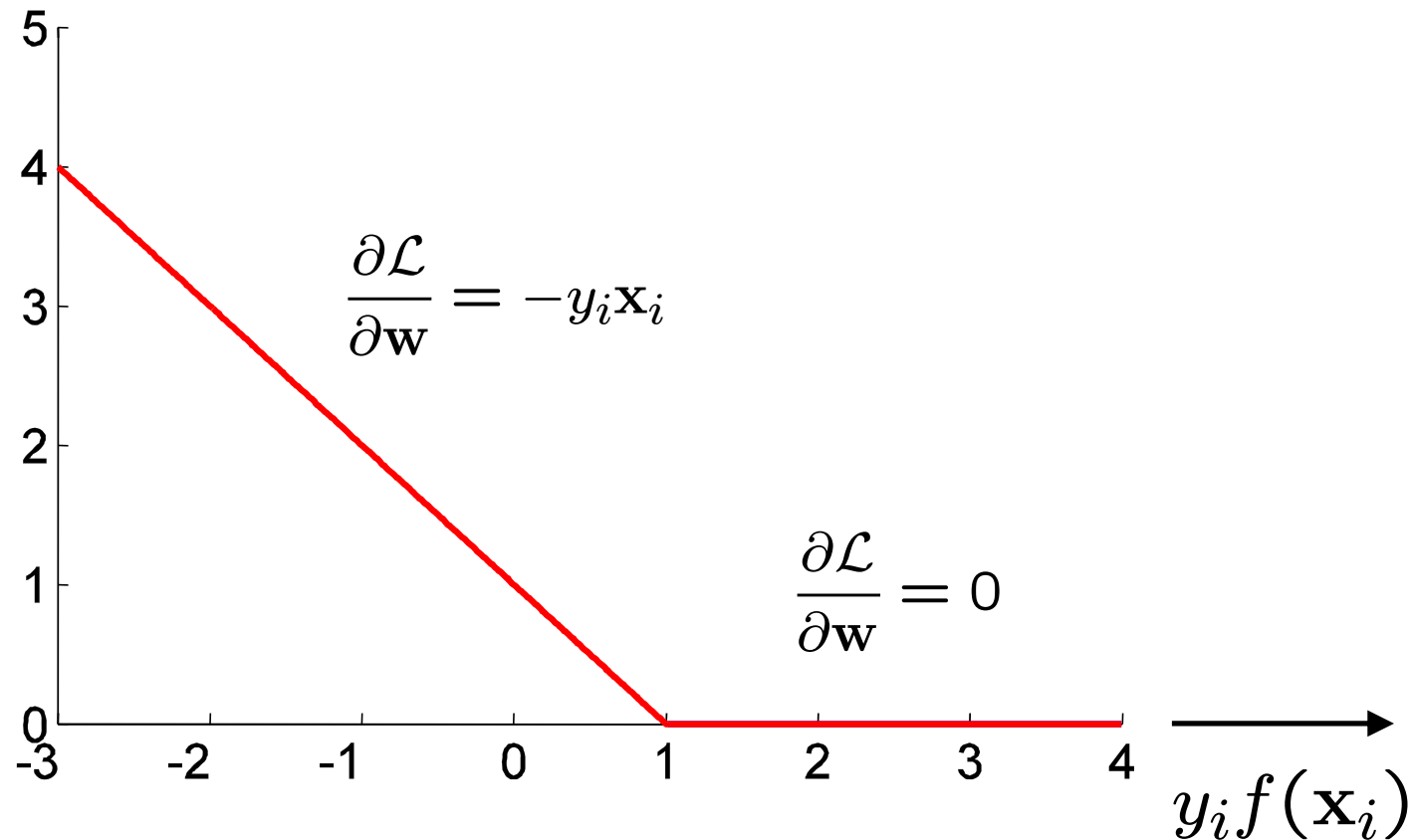
$$\begin{aligned} \min_{\mathbf{w}} \mathcal{C}(\mathbf{w}) &= \frac{\lambda}{2} \|\mathbf{w}\|^2 + \frac{1}{N} \sum_i^N \max(0, 1 - y_i f(\mathbf{x}_i)) \\ &= \frac{1}{N} \sum_i^N \left(\frac{\lambda}{2} \|\mathbf{w}\|^2 + \max(0, 1 - y_i f(\mathbf{x}_i)) \right) \end{aligned}$$

(with $\lambda = 2/(NC)$ up to an overall scale of the problem) and $f(\mathbf{x}) = \mathbf{w}^\top \mathbf{x} + b$

Because the hinge loss is not differentiable, a **sub-gradient** is computed

Sub-gradient for hinge loss

$$\mathcal{L}(\mathbf{x}_i, y_i; \mathbf{w}) = \max(0, 1 - y_i f(\mathbf{x}_i)) \quad f(\mathbf{x}_i) = \mathbf{w}^\top \mathbf{x}_i + b$$



Sub-gradient descent algorithm for SVM

$$\mathcal{C}(\mathbf{w}) = \frac{1}{N} \sum_i^N \left(\frac{\lambda}{2} \|\mathbf{w}\|^2 + \mathcal{L}(\mathbf{x}_i, y_i; \mathbf{w}) \right)$$

The iterative update is

$$\begin{aligned} \mathbf{w}_{t+1} &\leftarrow \mathbf{w}_t - \eta \nabla_{\mathbf{w}_t} \mathcal{C}(\mathbf{w}_t) \\ &\leftarrow \mathbf{w}_t - \eta \frac{1}{N} \sum_i^N (\lambda \mathbf{w}_t + \nabla_{\mathbf{w}} \mathcal{L}(\mathbf{x}_i, y_i; \mathbf{w}_t)) \end{aligned}$$

where η is the learning rate.


Then each iteration t involves cycling through the training data with the updates:

$$\begin{aligned} \mathbf{w}_{t+1} &\leftarrow \mathbf{w}_t - \eta(\lambda \mathbf{w}_t - y_i \mathbf{x}_i) && \text{if } y_i f(\mathbf{x}_i) < 1 \\ &\leftarrow \mathbf{w}_t - \eta \lambda \mathbf{w}_t && \text{otherwise} \end{aligned}$$

In the Pegasos algorithm the learning rate is set at $\eta_t = \frac{1}{\lambda t}$

Background reading and more ...

- Next lecture – see that the SVM can be expressed as a sum over the support vectors:

$$f(x) = \sum_i \alpha_i y_i (\mathbf{x}_i^\top \mathbf{x}) + b$$


support vectors

- On web page:
<http://www.robots.ox.ac.uk/~az/lectures/ml>
- links to SVM tutorials and video lectures
- MATLAB SVM demo