

Machine Learning 10-601

Tom M. Mitchell
Machine Learning Department
Carnegie Mellon University

November 8, 2017

Today: Kernel methods, SVM

- Support Vector Machines
- Margin-based classifiers

Readings:

Recommended:
Kernels: Bishop Ch. 6.1
SVMs: Bishop Ch. 7, through 7.1.2

Optional:
Bishop Ch 6.2, 6.3

Thanks to Aarti Singh, Eric Xing, Maria Balcan,
John Shawe-Taylor for several slides

Linear Regression: Dual Form

Primal form:

Learn $\hat{f}(\mathbf{x}) = \sum_{i=1}^N x_i w_i = \langle \mathbf{x}, \mathbf{w} \rangle = \mathbf{x}^T \mathbf{w}$

$$\mathbf{w} = \arg \min_{\mathbf{w}} \|\mathbf{y} - \mathbf{X}\mathbf{w}\|^2 + \lambda \|\mathbf{w}\|^2$$

Solution: $\mathbf{w} = (\mathbf{X}^T \mathbf{X} + \lambda \mathbf{I})^{-1} \mathbf{X}^T \mathbf{y}$

Dual form: use fact that

$$\mathbf{w} = \sum_{l=1}^M \alpha_l \mathbf{x}^l$$

Learn $\hat{f}(\mathbf{x}) = \sum_{l=1}^M \alpha_l \langle \mathbf{x}, \mathbf{x}^l \rangle$

Solution: $\alpha = (\mathbf{X} \mathbf{X}^T + \lambda \mathbf{I})^{-1} \mathbf{y}$

Key idea: Dual form expresses linear regression $f(x)$ in terms of dot products with training examples.

- more efficient computation when features per example > # exams
- turns training and application of learned model into dot products, which allows using kernel functions

Kernels : Key Points

- Many learning tasks are framed as optimization problems
- Primal and Dual formulations of optimization problems
- Dual version framed in terms of dot products between x 's
- Kernel functions $k(x,y)$ allow calculating dot products $\langle \Phi(x), \Phi(y) \rangle$ without bothering to project x into $\Phi(x)$
- Leads to major efficiencies, and ability to use very high dimensional (virtual) feature spaces

Example: Quadratic Kernel

Suppose we have data originally in 2D, but project it into 3D using $\Phi(\mathbf{x})$

$$\mathbf{x} = \begin{bmatrix} x_1 \\ x_2 \end{bmatrix} \quad \underline{\Phi(\mathbf{x})} = \begin{bmatrix} x_1^2 \\ \sqrt{2}x_1x_2 \\ x_2^2 \end{bmatrix}$$

$$Y = w_1 x_1 + w_2 x_2$$

$$Y = w_1 x_1^2 + w_2 x_1 x_2 + w_3 x_2^2$$

this converts our original linear regression into quadratic regression!

But we can use the following kernel function to calculate inner products in the projected 3D space, in terms of operations in the 2D space

$$\langle \Phi(\mathbf{x}_i), \Phi(\mathbf{x}_j) \rangle = \langle \mathbf{x}_i, \mathbf{x}_j \rangle^2 \equiv \kappa_{\Phi}(\mathbf{x}_i, \mathbf{x}_j)$$

And use it to train and apply our regression function, never leaving 2D space

$$\hat{f}(\mathbf{x}) = \sum_{l=1}^M \alpha_l \kappa(\mathbf{x}, \mathbf{x}^l) \quad \alpha = (\mathbf{K} + \lambda \mathbf{I})^{-1} \mathbf{y} \quad \mathbf{K}_{i,j} = \kappa(\mathbf{x}_i, \mathbf{x}_j)$$

Which Functions Can Be Kernels?

- not all functions
- for some definitions of $k(x_1, x_2)$ there is no corresponding projection $\phi(x)$
- Nice theory on this, including how to construct new kernels from existing ones

Some Common Mercer Kernels

- Polynomials of degree d

$$K(\mathbf{u}, \mathbf{v}) = (\mathbf{u} \cdot \mathbf{v})^d$$

- Polynomials of degree up to d

$$K(\mathbf{u}, \mathbf{v}) = (\mathbf{u} \cdot \mathbf{v} + 1)^d$$

- Gaussian/Radial kernels (polynomials of all orders – projected space has infinite dimension)

$$K(\mathbf{u}, \mathbf{v}) = \exp\left(-\frac{\|\mathbf{u} - \mathbf{v}\|^2}{2\sigma^2}\right)$$

- String kernels: \mathbf{u} and \mathbf{v} are strings over vocabulary A

$$K(u, v) = \sum_{s \in A^*} w_s c(s, u) c(s, v)$$

$c(s, x)$ = number of times substring s occurs in x

Properties of Kernels

Theorem (Mercer)

K is a kernel if and only if:

- K is symmetric
- For any set of training points x_1, x_2, \dots, x_m and for any $a_1, a_2, \dots, a_m \in R$, we have:

$$\sum_{i,j} a_i a_j K(x_i, x_j) \geq 0$$

$$a^T K a \geq 0$$

I.e., $K = (K(x_i, x_j))_{i,j=1,\dots,n}$ is positive semi-definite.

Kernel, Closure Properties

Easily create new kernels using basic ones!



Fact: If $K_1(\cdot, \cdot)$ and $K_2(\cdot, \cdot)$ are kernels $c_1 \geq 0, c_2 \geq 0$,
then $K(x, z) = c_1 K_1(x, z) + c_2 K_2(x, z)$ is a kernel.

Key idea: concatenate the ϕ spaces.

$$\phi(x) = (\sqrt{c_1} \phi_1(x), \sqrt{c_2} \phi_2(x))$$

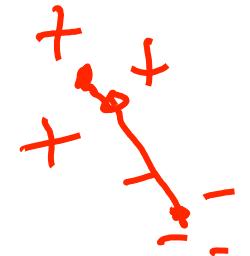
$$\phi(x) \cdot \phi(z) = c_1 \phi_1(x) \cdot \phi_1(z) + c_2 \phi_2(x) \cdot \phi_2(z)$$

$$K_1(x, z)$$

$$K_2(x, z)$$

Simple Kernel Based Classifier

- Consider finding the centres of mass of positive and negative examples and classifying a test point by measuring which is closest



$$h(\mathbf{x}) = \text{sgn} (\|\phi(\mathbf{x}) - \phi_{S_-}\|^2 - \|\phi(\mathbf{x}) - \phi_{S_+}\|^2)$$

- we can express as a function of kernel evaluations

$$h(\mathbf{x}) = \text{sgn} \left(\frac{1}{m_+} \sum_{i=1}^{m_+} \kappa(\mathbf{x}, \mathbf{x}_i) - \frac{1}{m_-} \sum_{i=m_++1}^m \kappa(\mathbf{x}, \mathbf{x}_i) - b \right),$$

where

$$b = \frac{1}{2m_+^2} \sum_{i,j=1}^{m_+} \kappa(\mathbf{x}_i, \mathbf{x}_j) - \frac{1}{2m_-^2} \sum_{i,j=m_++1}^m \kappa(\mathbf{x}_i, \mathbf{x}_j)$$

[slide from John Shawe-Taylor]

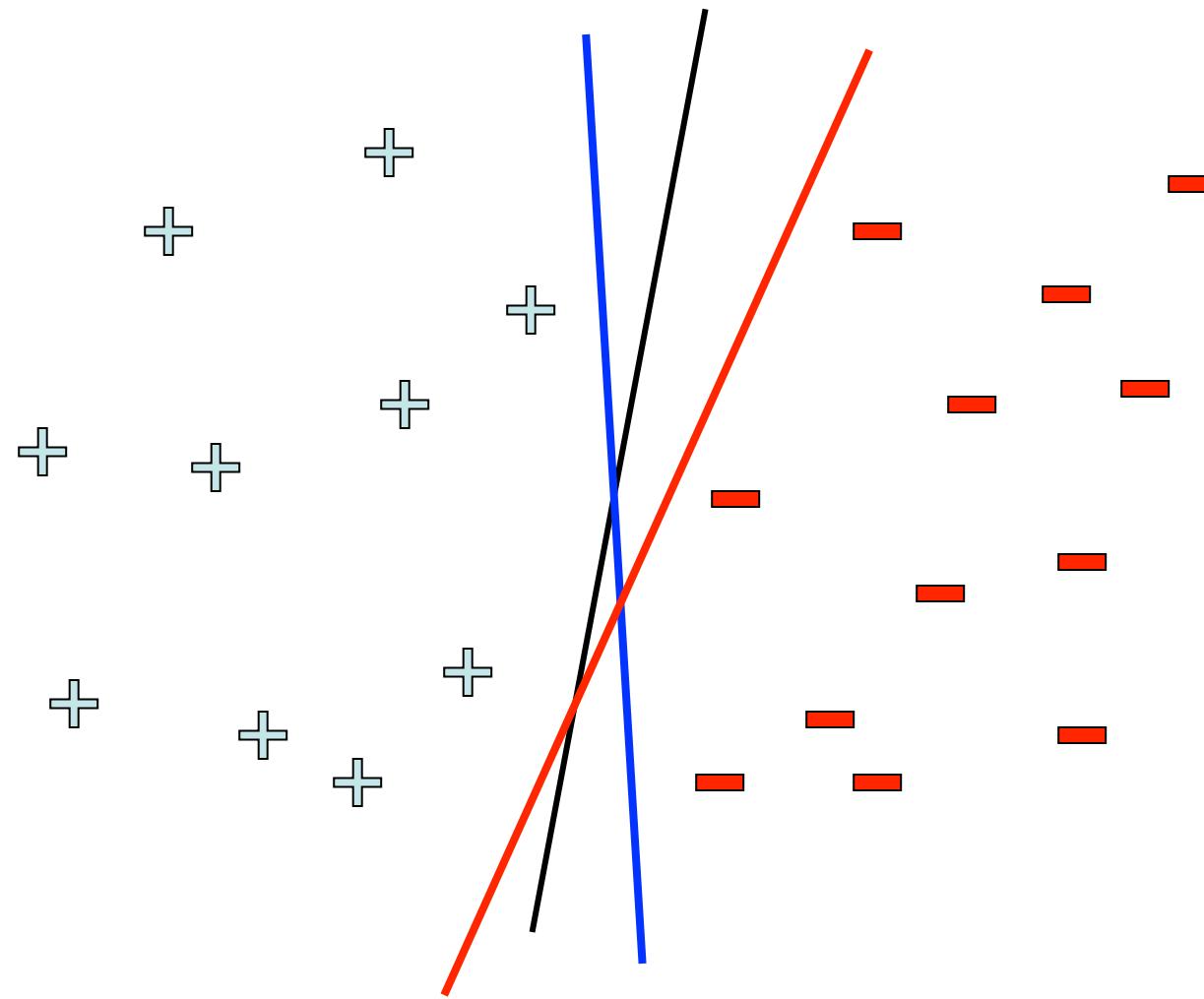
Support Vector Machines

Support Vector Machines

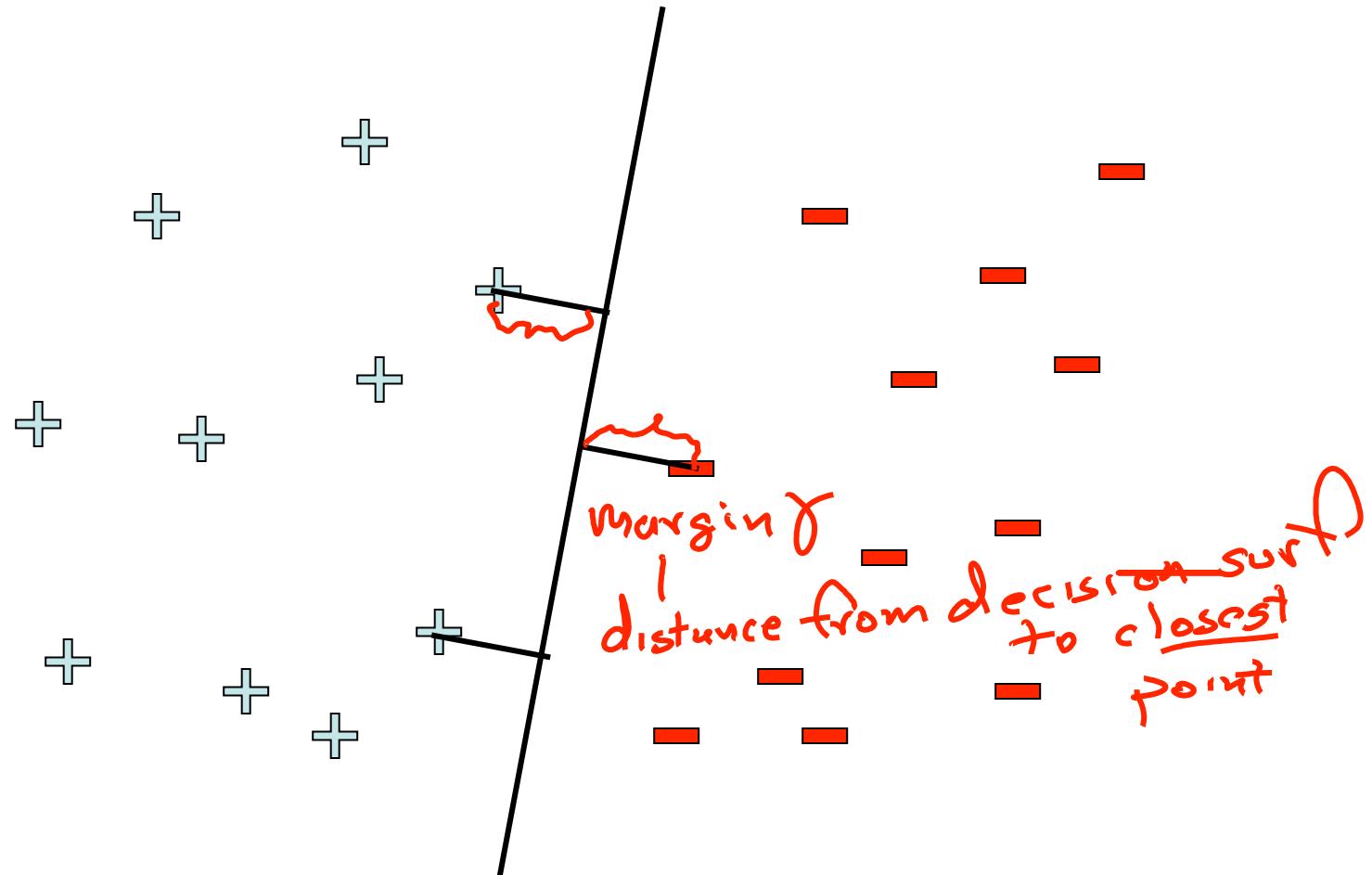
Two key ideas:

1. new loss function (maximize margin)
2. use kernels to achieve non-linear boundaries

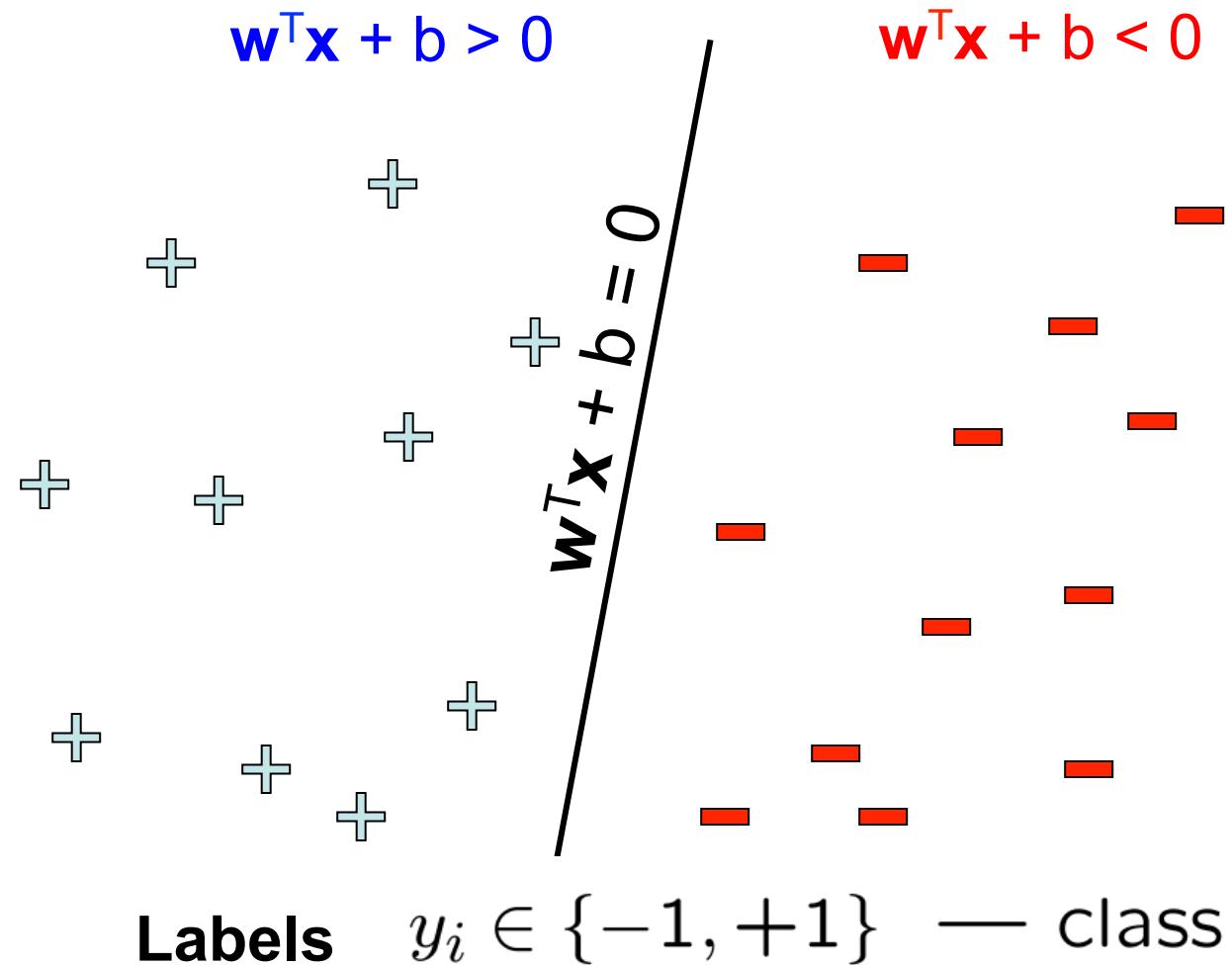
Linear classifiers – which line is better?



Pick the one with the largest margin!

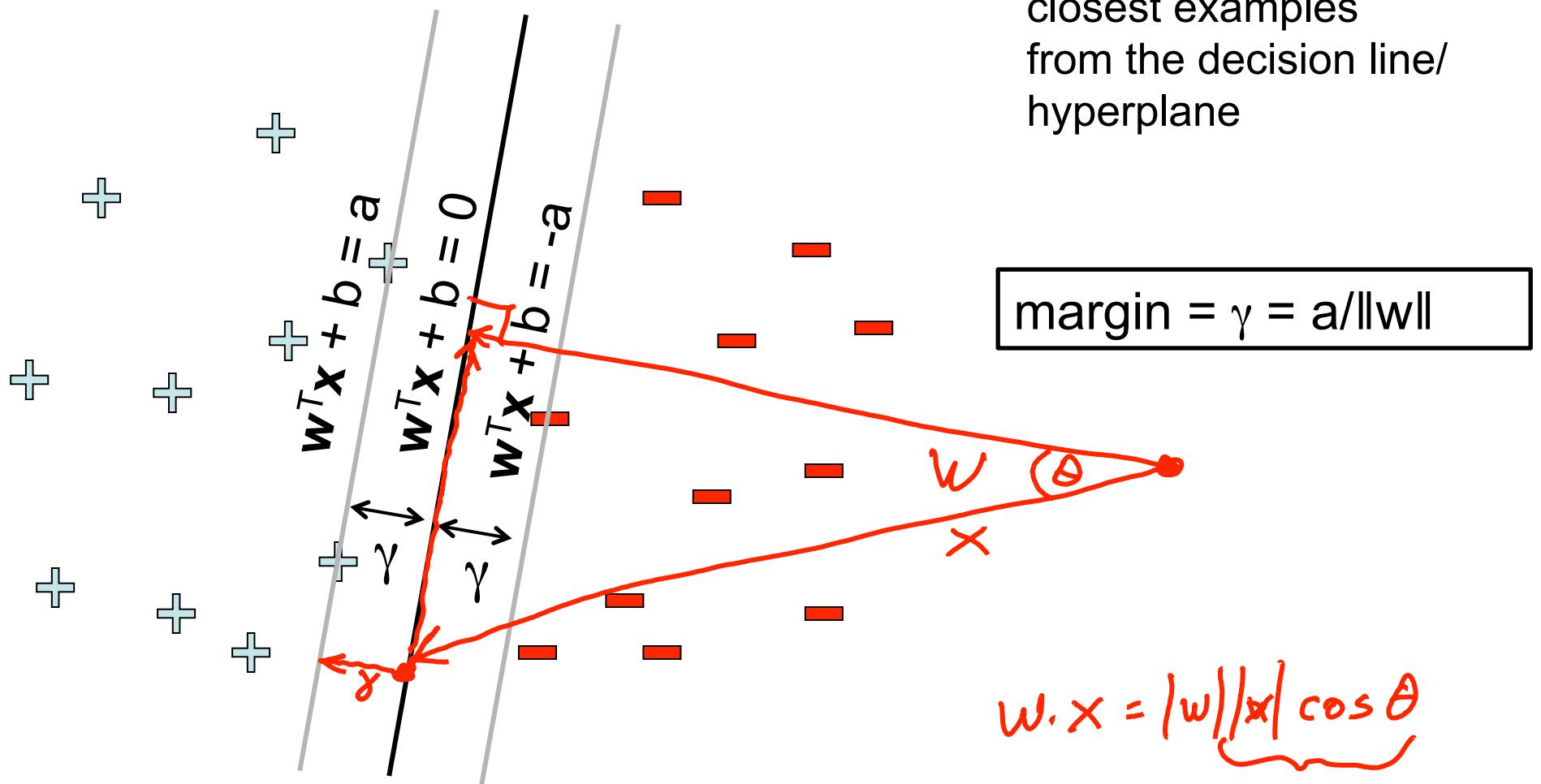


Parameterizing the decision boundary

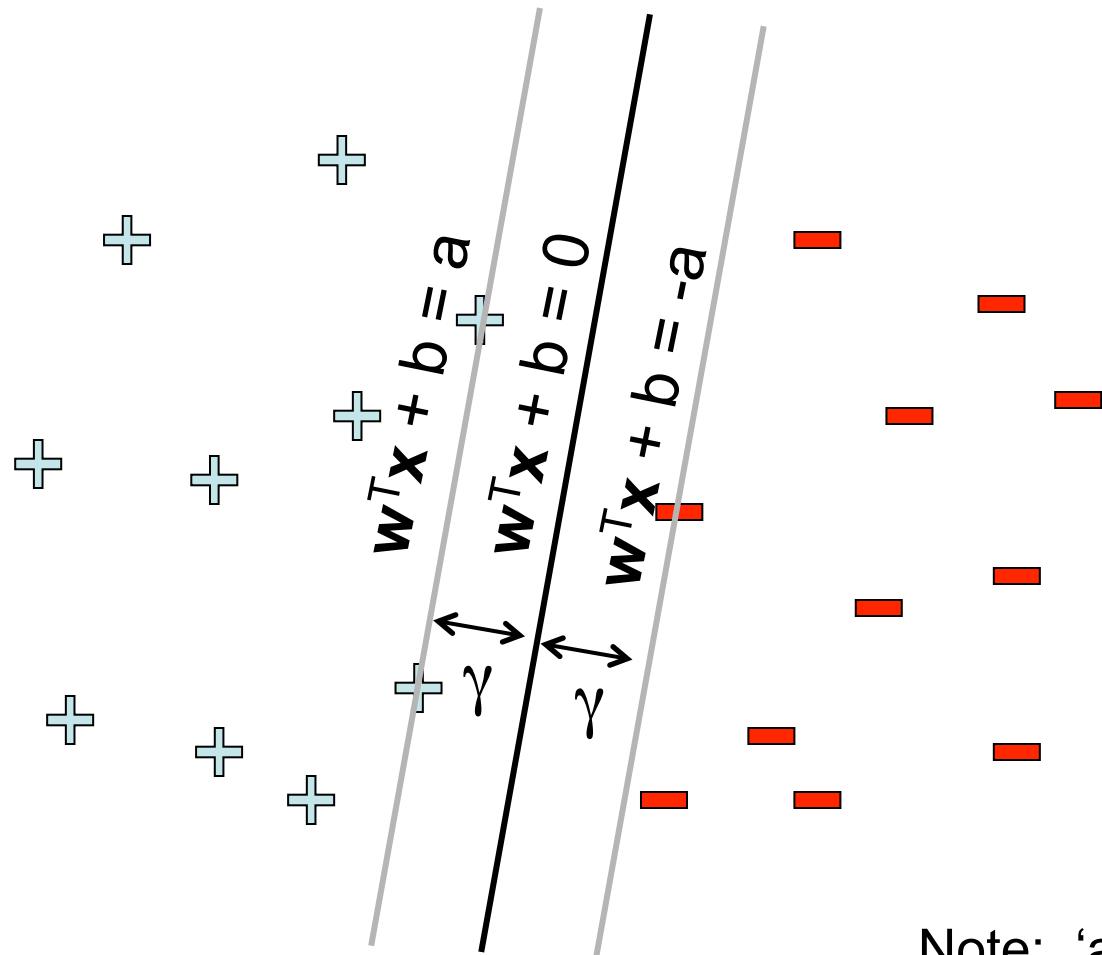


SVM: Maximize the margin

Margin = Distance of
closest examples
from the decision line/
hyperplane



Maximizing the margin



Margin = Distance of
closest examples
from the decision line/
hyperplane

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

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

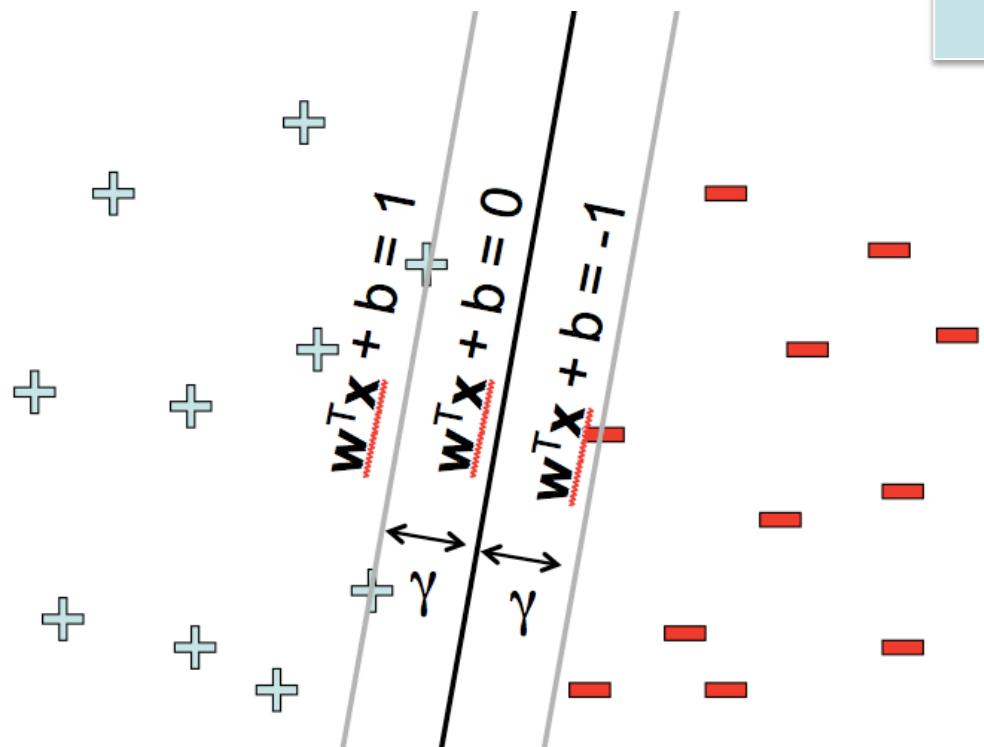
$$\text{s.t. } (w^T x_j + b) y_j \geq a \quad \forall j$$

Note: 'a' is arbitrary (can normalize
equations by a)

Support Vector Machine (primal form)

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

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



Primal form:

$$\min_{\mathbf{w}, b} \mathbf{w}^T \mathbf{w}$$

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

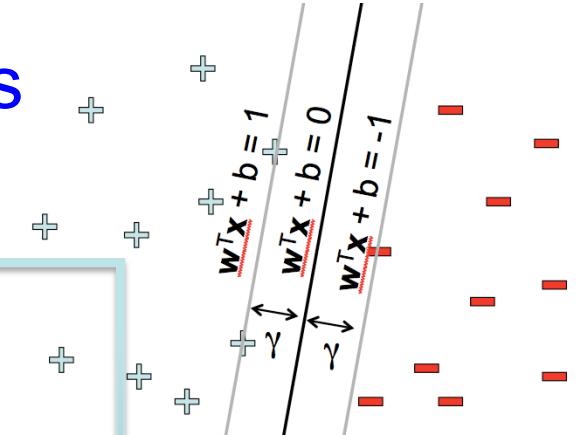
Solve efficiently by quadratic programming (QP)
– Well-studied solution algorithms

We can solve either primal or dual forms

Primal form: solve for \mathbf{w}, b

$$\begin{aligned} \min_{\mathbf{w}, b} \quad & \mathbf{w}^T \mathbf{w} \\ \text{s.t.} \quad & y_l (\mathbf{w}^T \mathbf{x}_l + b) \geq 1 \quad \forall l \in \text{training examples} \end{aligned}$$

Classification test for new \mathbf{x} : $\mathbf{w}^T \mathbf{x} + b > 0$



Dual form: solve for $\alpha_1 \dots \alpha_M$

$$\max_{\alpha_1 \dots \alpha_M} \sum_{l=1}^M \alpha_l - \frac{1}{2} \sum_{j=1}^M \sum_{k=1}^M \alpha_j \alpha_k y_j y_k \langle \mathbf{x}_j, \mathbf{x}_k \rangle$$

$$\text{s.t.} \quad \alpha_l \geq 0 \quad \forall l \in \text{training examples}$$

$$\sum_{l=1}^M \alpha_l y_l = 0$$

Classification test for new \mathbf{x} : $\sum_{l \in \text{SV's}} \alpha_l y_l \langle \mathbf{x}, \mathbf{x}_l \rangle + b > 0$

both are QP problems with a single local optimum!

What is quadratic programming?

A way to solve optimization problems of the form:

$$\begin{aligned} \text{Minimize } f(\mathbf{x}) &= \mathbf{c}\mathbf{x} + \frac{1}{2} \mathbf{x}^T \mathbf{Q} \mathbf{x} \\ \text{subject to } &\mathbf{A}\mathbf{x} \leq \mathbf{b} \text{ and } \mathbf{x} \geq \mathbf{0} \end{aligned}$$

Where \mathbf{Q} is symmetric matrix.

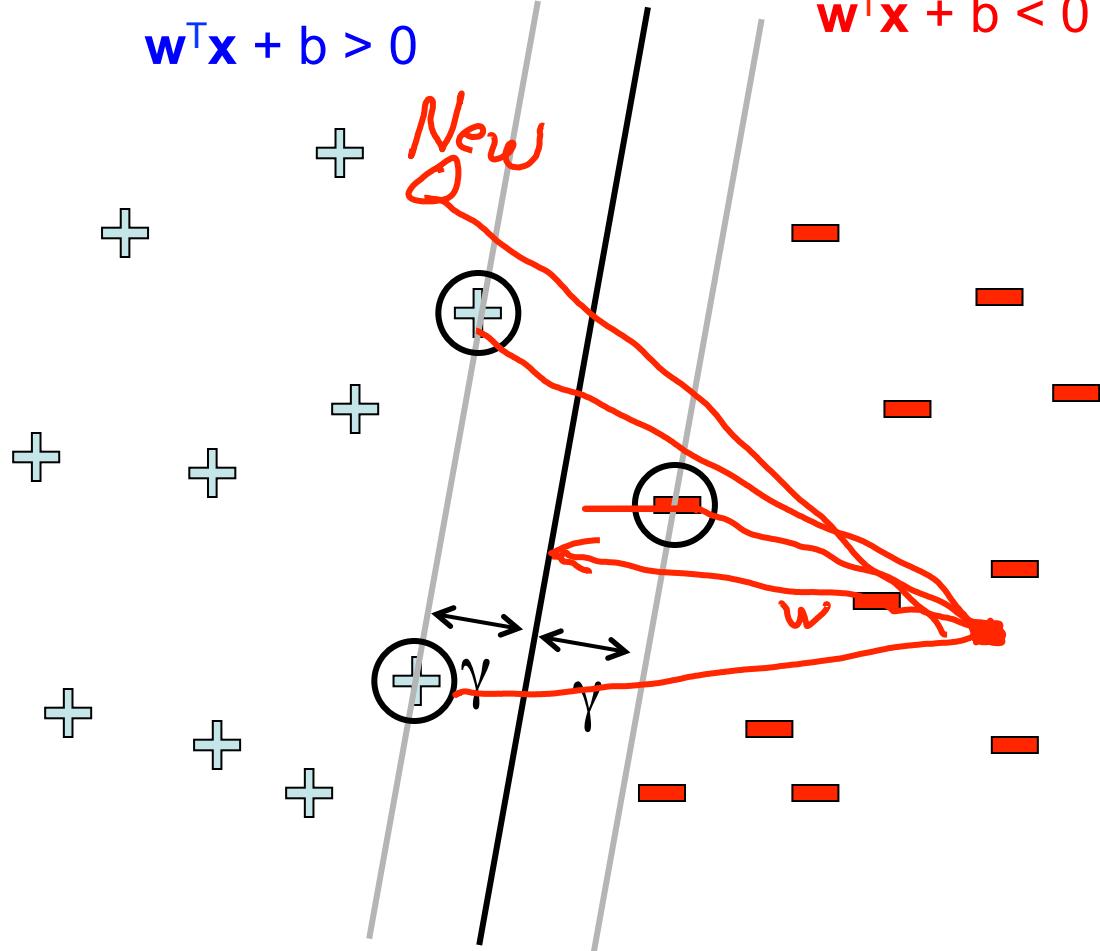
If $f(\mathbf{x})$ is strictly convex for all feasible points (e.g., \mathbf{Q} is positive semi-definite)

then there is only one local minimum, which is global min.

Support Vectors

$$\sum_{l \in SV, S} \alpha_l y_l \langle \mathbf{x}, \mathbf{x}_l \rangle + b > 0$$

$$\sum_{l \in SV, S} \alpha_l y_l \langle \mathbf{x}, \mathbf{x}_l \rangle + b < 0$$



Linear hyperplane is fully 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?

$$\leq d+1$$

* KKT conditions on optimization problem assure other $\alpha_i = 0$

Kernel SVM

And because the dual form depends only on inner products, we can apply the kernel trick to work in a (virtual) projected space $\Phi : X \rightarrow F$

Primal form: solve for \mathbf{w}, b in the projected higher dim. space

$$\min_{\mathbf{w}, b} \quad \mathbf{w}^T \mathbf{w}$$

$$\text{s.t. } y_l(\mathbf{w}^T \Phi(\mathbf{x}_l) + b) \geq 1 \quad \forall l \in \text{ training examples}$$

Classification test for new \mathbf{x} $\mathbf{w}^T \Phi(\mathbf{x}) + b > 0$

Dual form: solve for $\alpha_1 \dots \alpha_M$ in the original low dim. space

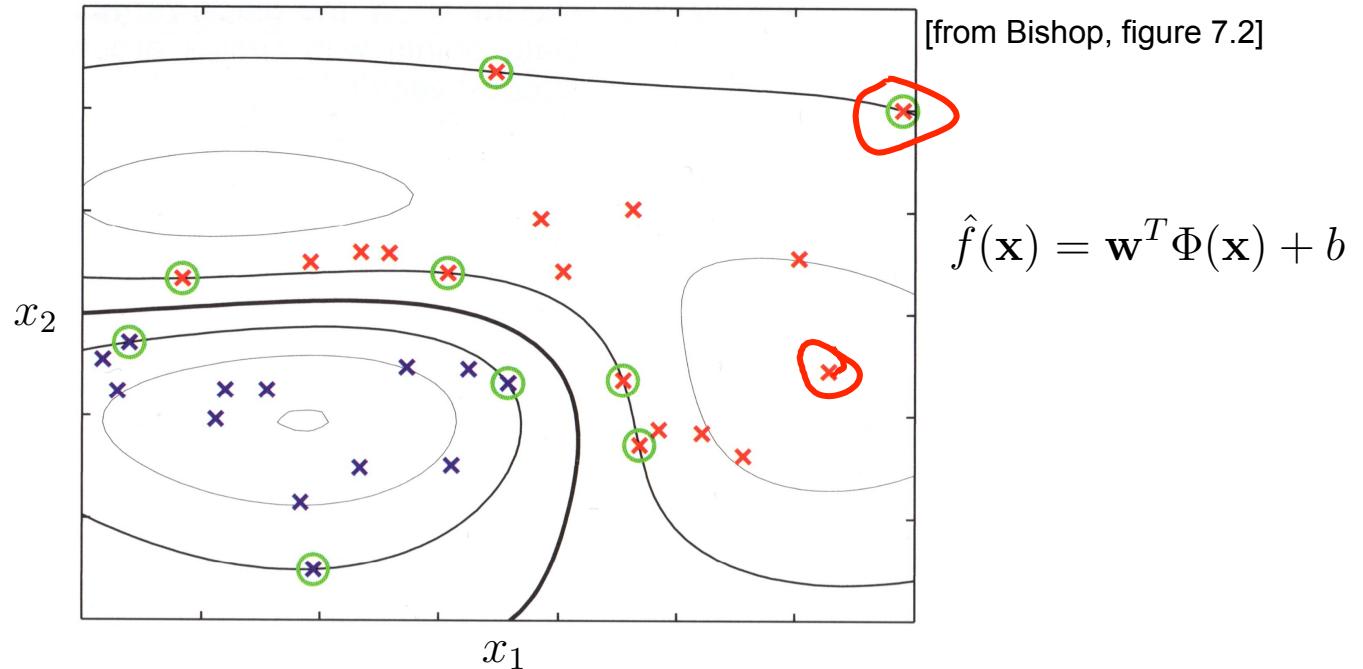
$$\max_{\alpha_1 \dots \alpha_M} \quad \sum_{l=1}^M \alpha_l - \frac{1}{2} \sum_{j=1}^M \sum_{k=1}^M \alpha_j \alpha_k y_j y_k \kappa(\mathbf{x}_j, \mathbf{x}_k)$$

$$\text{s.t. } \alpha_l \geq 0 \quad \forall l \in \text{ training examples}$$

$$\sum_{l=1}^M \alpha_l y_l = 0$$

Classification test for new \mathbf{x} : $\sum_{l \in \text{SV's}} \alpha_l y_l \kappa(\mathbf{x}, \mathbf{x}_l) + b > 0$

SVM Decision Surface using Gaussian Kernel



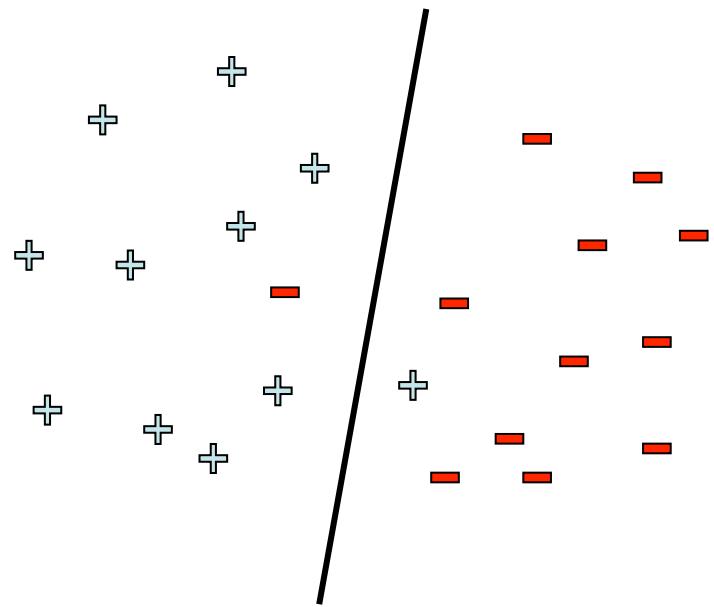
Circled points are the support vectors: training examples with non-zero α_l

Points plotted in original 2-D space.

Contour lines show constant $\hat{f}(\mathbf{x})$

$$\hat{f}(\mathbf{x}) = b + \sum_{l=1}^M \alpha_l y_l \kappa(\mathbf{x}, \mathbf{x}_l) = b + \sum_{l=1}^M \alpha_l y_l \exp(-\|\mathbf{x} - \mathbf{x}_l\|^2 / 2\sigma^2)$$

What if data is not linearly separable?



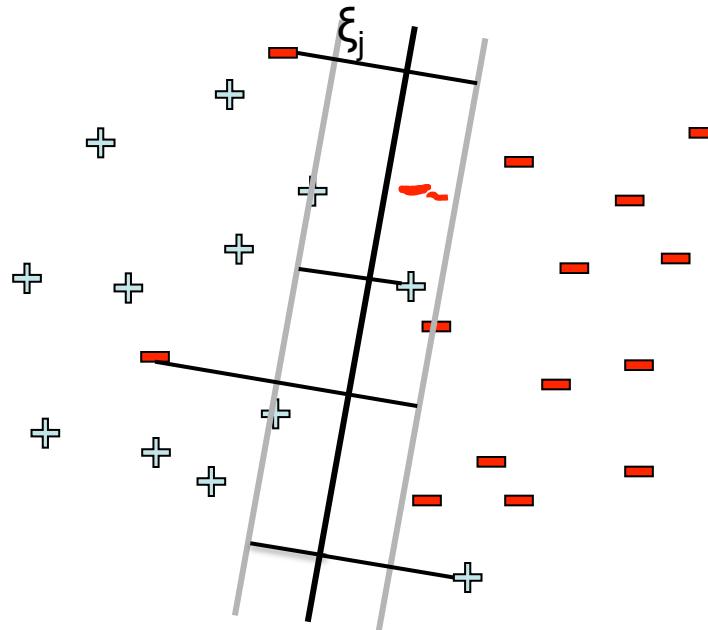
Use features of features
of features of features....

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

But run risk of overfitting!

Support Vector Machine with soft margins

Allow “error” in classification



Soft margin approach

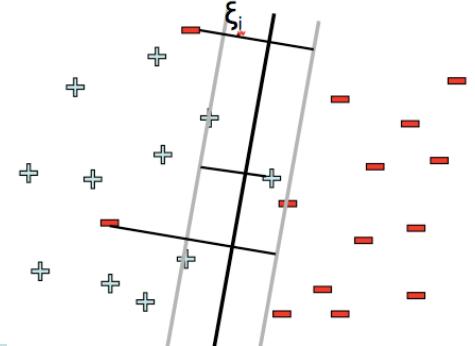
$$\begin{aligned} & \min_{\mathbf{w}, b} \mathbf{w}^T \mathbf{w} + C \sum_j \xi_j \\ \text{s.t. } & (\mathbf{w}^T \mathbf{x}_j + b) y_j \geq 1 - \xi_j \quad \forall j \\ & \xi_j \geq 0 \quad \forall j \end{aligned}$$

ξ_j - “slack” variables
= (>1 if x_j misclassified)
pay linear penalty if mistake

C - tradeoff parameter (chosen by cross-validation)

Still QP ☺

Primal and Dual Forms for Soft Margin SVM



Primal form: solve for \mathbf{w}, b in the projected higher dim. space

$$\min_{\mathbf{w}, b} \quad \frac{1}{2} \mathbf{w}^T \mathbf{w} + C \sum_{l=1}^M \xi_l$$

$$\text{s.t. } y_l (\mathbf{w}^T \Phi(\mathbf{x}_l) + b) \geq 1 - \xi_l \quad \forall l \in \text{training examples}$$

$$\xi_l \geq 0 \quad \forall l \in \text{training examples}$$

Dual form: solve for $\alpha_1 \dots \alpha_M$ in the original low dim. space

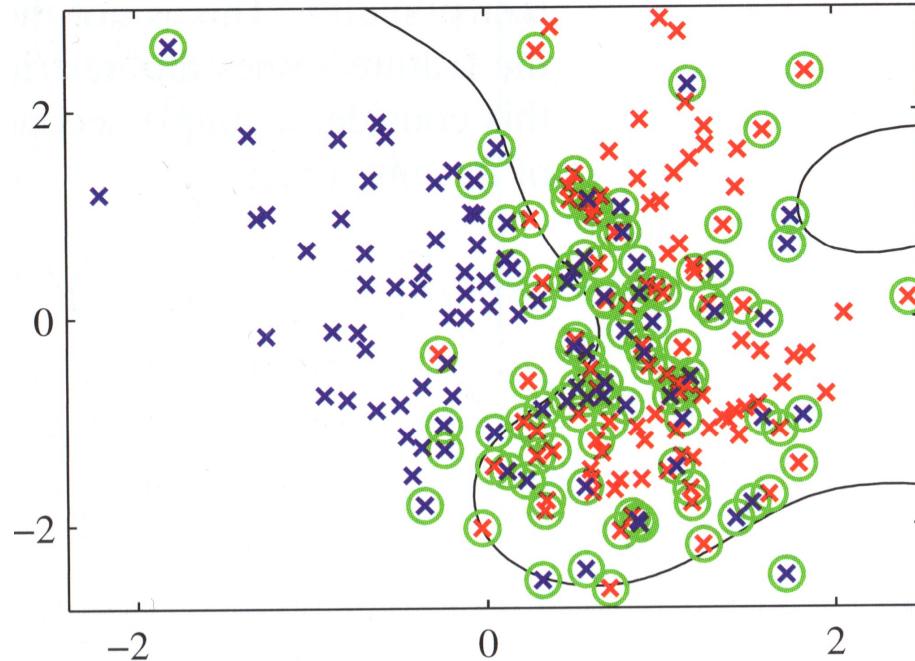
$$\max_{\alpha_1 \dots \alpha_M} \quad \sum_{l=1}^M \alpha_l - \frac{1}{2} \sum_{j=1}^M \sum_{k=1}^M \alpha_j \alpha_k y_j y_k \kappa(\mathbf{x}_j, \mathbf{x}_k)$$

$$\text{s.t. } 0 \leq \alpha_l \leq C \quad \forall l \in \text{training examples}$$

$$\sum_{l=1}^M \alpha_l y_l = 0$$

both are QP problems with a single local optimum ☺

SVM Soft Margin Decision Surface using Gaussian Kernel



[from Bishop, figure 7.4]

$$\hat{f}(\mathbf{x}) = \mathbf{w}^T \Phi(\mathbf{x}) + b$$

Circled points are the support vectors: training examples with non-zero α_l

Points plotted in original 2-D space.

Contour lines show constant $\hat{f}(\mathbf{x})$

$$\hat{f}(\mathbf{x}) = b + \sum_{l=1}^M \alpha_l y_l \kappa(\mathbf{x}, \mathbf{x}_l) = b + \sum_{l=1}^M \alpha_l y_l \exp(-\|\mathbf{x} - \mathbf{x}_l\|^2 / 2\sigma^2)$$

SVM Summary

- Objective: maximize margin between decision surface and data
- Primal and dual formulations
 - dual represents classifier decision in terms of *support vectors*
- Kernel SVM's
 - learn linear decision surface in high dimension space, working in original low dimension space
- Handling noisy data: soft margin “slack variables”
 - again primal and dual forms
- SVM algorithm: Quadratic Program optimization
 - single global optimum

Extra optional slides we did not cover

Maximizing Margin as an Objective Function

- We've talked about many learning algorithms, with different objective functions
 - 0-1 loss
 - sum sq error
 - maximum log data likelihood
 - MAP
 - maximum margin

How are these all related?

Slack variables – Hinge loss

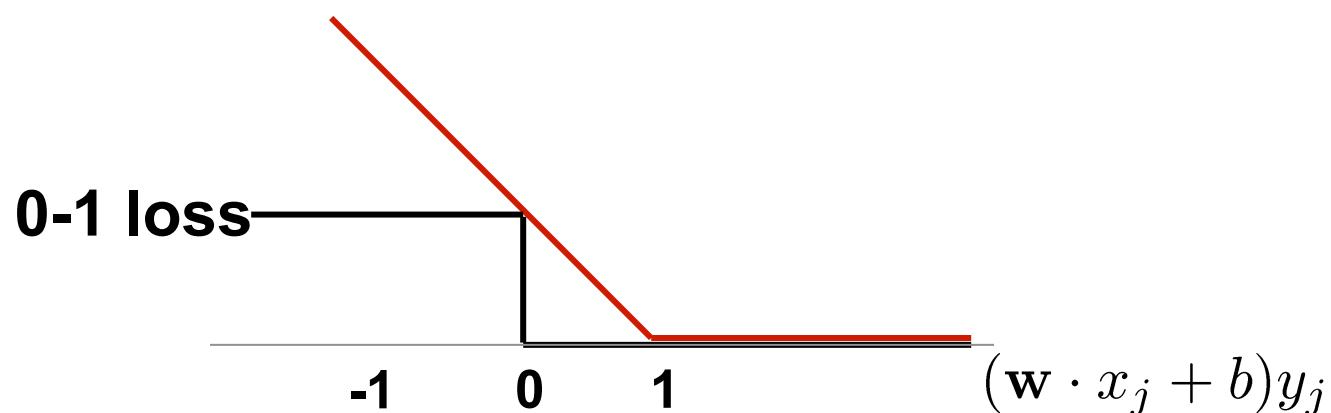
Complexity penalization

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

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

$$\begin{aligned} & \min_{\mathbf{w}, b} \mathbf{w}^T \mathbf{w} + C \sum_j \xi_j \\ & \text{s.t. } (\mathbf{w}^T \mathbf{x}_j + b) y_j \geq 1 - \xi_j \quad \forall j \\ & \quad \xi_j \geq 0 \quad \forall j \end{aligned}$$

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



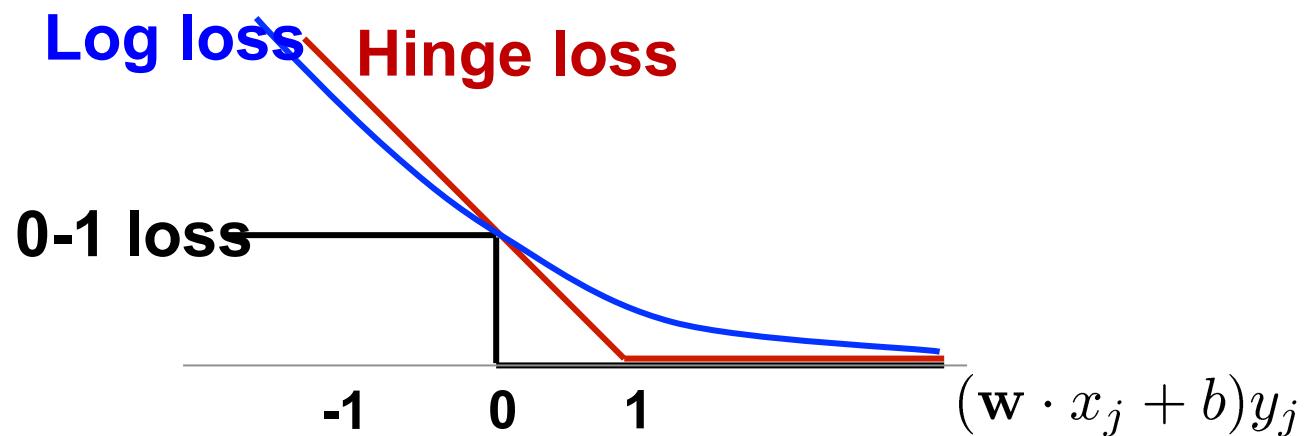
SVM vs. Logistic Regression

SVM : **Hinge loss**

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

Logistic Regression : **Log loss** (negative log conditional likelihood)

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



SVM: PAC Results?

VC dimension: examples

What is VC dimension of

- $H_2 = \{ ((w_0 + w_1x_1 + w_2x_2) > 0 \rightarrow y=1) \}$
 - $VC(H_2)=3$
- For H_n = linear separating hyperplanes in n dimensions, $VC(H_n)=n+1$



$$m \geq \frac{1}{\epsilon} (4 \log_2(2/\delta) + 8VC(H) \log_2(13/\epsilon))$$

Margin-based PAC Results

[Shawe-Taylor, Langford, McCallester]

Consider a fixed distribution D on pairs $\langle x, y \rangle$ with $x \in R^d$ satisfying $\|x\| = 1$ and $y \in \{-1, 1\}$. We are interested in finding a weight vector w with $\|w\| = 1$ such that the sign of $w \cdot x$ predicts y . For $\gamma > 0$ the error rate of w on distribution D relative to safety margin γ , denoted $\ell_\gamma(w, D)$ is defined as follows.

$$\ell_\gamma(w, D) = P_{\langle x, y \rangle \sim D} [(w \cdot x)y \leq \gamma]$$

Let S be a sample of m pairs drawn IID from the distribution D . The sample S can be viewed as an empirical distribution on pairs. We are interested in bounding $\ell_0(w, D)$ in terms of $\ell_\gamma(w, S)$ and the margin γ . Bartlett and Shawe-Taylor use fat shattering arguments [2] to show that with probability at least $1 - \delta$ over the choice of the sample S we have the following simultaneously for all weight vectors w with $\|w\| = 1$ and margins $\gamma > 0$.

$$\ell_0(w, D) \leq \ell_\gamma(w, S) + 27.18 \sqrt{\frac{\log^2 m + 84}{m\gamma^2}} + O\left(\sqrt{\frac{\ln \frac{1}{\delta}}{m}}\right) \quad (1)$$

recall:

$$error_{true}(h) < error_{train}(h) + \sqrt{\frac{VC(H)(\ln \frac{2m}{VC(H)} + 1) + \ln \frac{4}{\delta}}{m}}$$