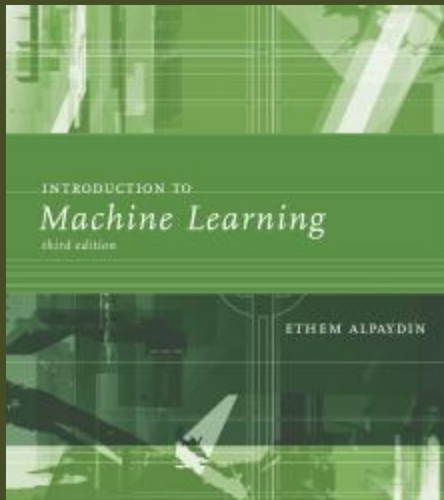
Lecture Slides for

# INTRODUCTION TO MACHINE LEARNING

3RD EDITION




ETHEM ALPAYDIN


*alpaydin@boun.edu.tr*
*http://www.cmpe.boun.edu.tr/~ethem/i2ml3e*

CHAPTER 13:

# KERNEL MACHINES

# Kernel Machines



- Discriminant-based: No need to estimate densities first
- Define the discriminant in terms of support vectors
- The use of kernel functions, application-specific measures of similarity
- No need to represent instances as vectors
- Convex optimization problems with a unique solution

# Optimal Separating Hyperplane



$$\mathcal{X} = \{\mathbf{x}^t, r^t\}_t \text{ where } r^t = \begin{cases} +1 & \text{if } \mathbf{x}^t \in C_1 \\ -1 & \text{if } \mathbf{x}^t \in C_2 \end{cases}$$

find $\mathbf{w}$ and $w_0$ such that

$$\mathbf{w}^T \mathbf{x}^t + w_0 \geq +1 \text{ for } r^t = +1$$

$$\mathbf{w}^T \mathbf{x}^t + w_0 \leq +1 \text{ for } r^t = -1$$

which can be rewritten as

$$r^t \left(\mathbf{w}^T \mathbf{x}^t + w_0\right) \geq +1$$

(Cortes and Vapnik, 1995; Vapnik, 1995)

# Margin

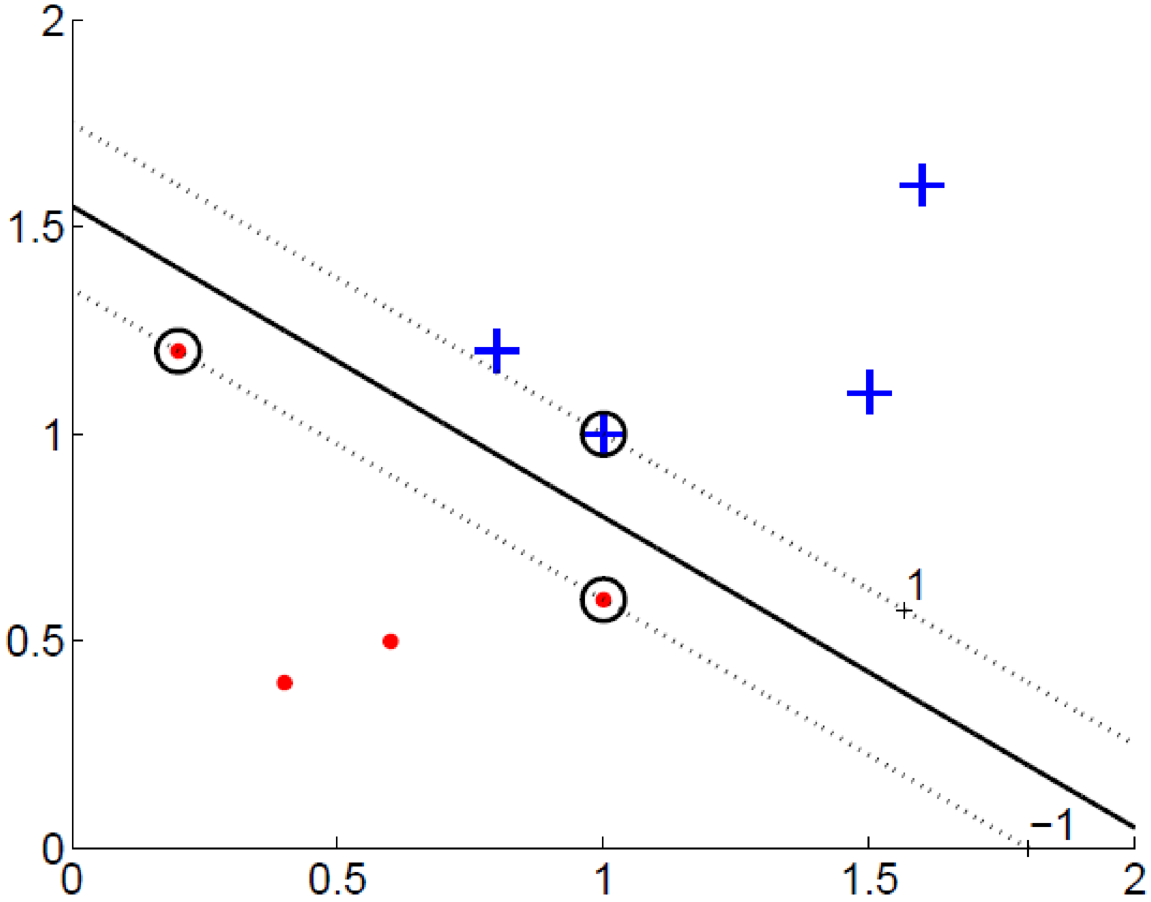


- Distance from the discriminant to the closest instances on either side
- Distance of x to the hyperplane is $\frac{\left|\mathbf{w}^T\mathbf{x}^t + w_0\right|}{\|\mathbf{w}\|}$
- We require $\frac{r^t\left(\mathbf{w}^T\mathbf{x}^t + w_0\right)}{\|\mathbf{w}\|} \geq \rho, \forall t$
- For a unique sol'n, fix $\rho\|w\|=1$, and to max margin

$$\min \frac{1}{2}\|\mathbf{w}\|^2 \text{ subject to } r^t\left(\mathbf{w}^T\mathbf{x}^t + w_0\right) \geq +1, \forall t$$

# Margin





$$\min \frac{1}{2}\|\mathbf{w}\|^2 \text{ subject to } r^t\left(\mathbf{w}^T\mathbf{x}^t + w_0\right) \geq +1, \forall t$$

$$L_p = \frac{1}{2}\|\mathbf{w}\|^2 - \sum_{t=1}^{N} \alpha^t \left[r^t\left(\mathbf{w}^T\mathbf{x}^t + w_0\right) - 1\right]$$

$$= \frac{1}{2}\|\mathbf{w}\|^2 - \sum_{t=1}^{N} \alpha^t r^t\left(\mathbf{w}^T\mathbf{x}^t + w_0\right) + \sum_{t=1}^{N} \alpha^t$$

$$\frac{\partial L_p}{\partial \mathbf{w}} = 0 \Rightarrow \mathbf{w} = \sum_{t=1}^{N} \alpha^t r^t \mathbf{x}^t$$

$$\frac{\partial L_p}{\partial w_0} = 0 \Rightarrow \sum_{t=1}^{N} \alpha^t r^t = 0$$

$$L_d = \frac{1}{2}\left(\mathbf{w}^T\mathbf{w}\right) - \mathbf{w}^T \sum_t \alpha^t r^t \mathbf{x}^t - w_0 \sum_t \alpha^t r^t + \sum_t \alpha^t$$

$$= -\frac{1}{2}\left(\mathbf{w}^T\mathbf{w}\right) + \sum_t \alpha^t$$

$$= -\frac{1}{2}\sum_t \sum_s \alpha^t \alpha^s r^t r^s \left(\mathbf{x}^t\right)^T \mathbf{x}^s + \sum_t \alpha^t$$

$$\text{subject to } \sum_{\text{t}} \alpha^t r^t = 0 \text{ and } \alpha^t \geq 0, \forall t$$

Most $\alpha^t$ are 0 and only a small number have $\alpha^t > 0$; they are the support vectors

# Soft Margin Hyperplane



- Not linearly separable

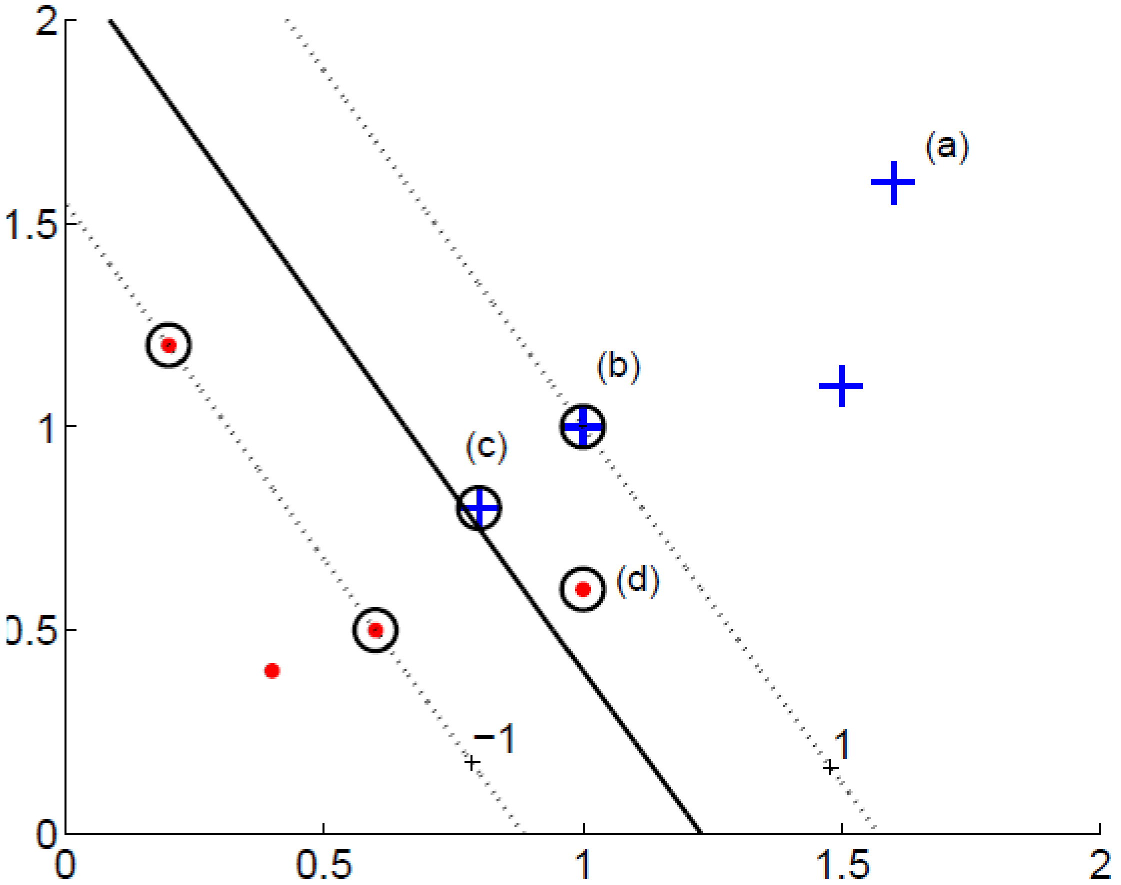
$$r^t\left(\mathbf{w}^T x^t + w_0\right) \geq 1 - \xi^t$$

- Soft error

$$\sum_t \xi^t$$

- New primal is

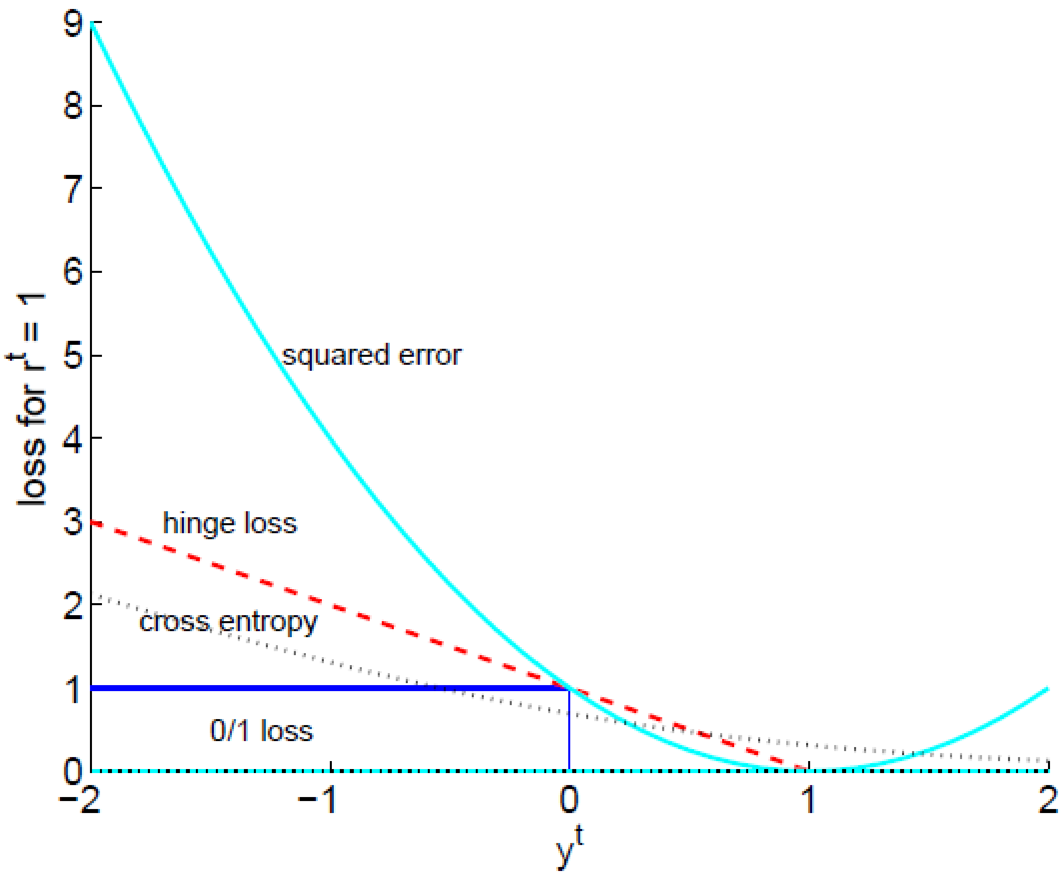
$$L_p = \frac{1}{2}\|\mathbf{w}\|^2 + C\sum_t \xi^t - \sum_t \alpha^t\left[r^t\left(\mathbf{w}^T x^t + w_0\right) - 1 + \xi^t\right] - \sum_t \mu^t \xi^t$$



(a)
(b)
(c)
(d)
−1
1
0
0.5
1
1.5
2

# Hinge Loss






$$: \begin{cases} 0 & \text{if } y^t r^t \geq 1 \\ 1 - y^t r^t & \text{otherwise} \end{cases}$$

# $\nu$-SVM



$$\min \frac{1}{2}\|\mathbf{w}\|^2 - \nu\rho + \frac{1}{N}\sum_t \xi^t$$

subject to

$$r^t\left(\mathbf{w}^T\mathbf{x}^t + w_0\right) \geq \rho - \xi^t, \xi^t \geq 0, \rho \geq 0$$

$$L_d = -\frac{1}{2}\sum_{t=1}^{N}\sum_s \alpha^t \alpha^s r^t r^s \left(x^t\right)^T x^s$$

subject to

$$\sum_t \alpha^t r^t = 0, 0 \leq \alpha^t \leq \frac{1}{N}, \sum_t \alpha^t \leq \nu$$

*$\nu$ controls the fraction of support vectors*

# Kernel Trick



- Preprocess input **x** by basis functions

$$\mathbf{z} = \boldsymbol{\varphi}(\mathbf{x}) \qquad g(\mathbf{z}) = \mathbf{w}^T\mathbf{z}$$

$$g(\mathbf{x}) = \mathbf{w}^T \boldsymbol{\varphi}(\mathbf{x})$$

- The SVM solution

$$\mathbf{w} = \sum_t \alpha^t r^t \mathbf{z}^t = \sum_t \alpha^t r^t \boldsymbol{\varphi}(\mathbf{x}^t)$$

$$g(\mathbf{x}) = \mathbf{w}^T \boldsymbol{\varphi}(\mathbf{x}) = \sum_t \alpha^t r^t \boxed{\boldsymbol{\varphi}(\mathbf{x}^t)^T \boldsymbol{\varphi}(\mathbf{x})}$$

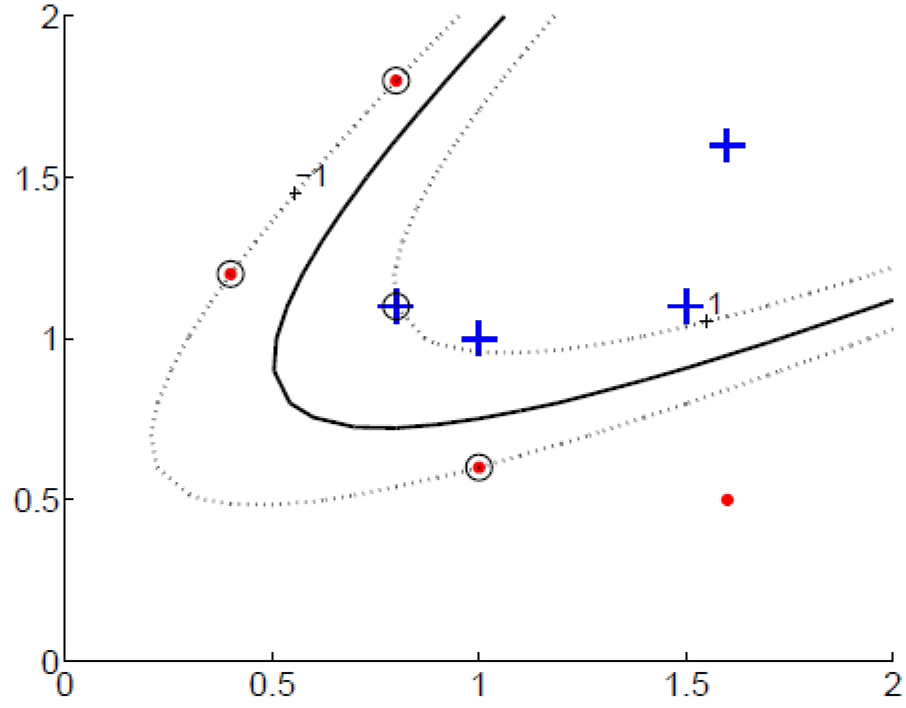
$$g(\mathbf{x}) = \sum_t \alpha^t r^t \boxed{K(\mathbf{x}^t, \mathbf{x})}$$

# Vectorial Kernels



- Polynomials of degree $q$:

$$K\left(\mathbf{x}^t,\mathbf{x}\right)=\left(\mathbf{x}^T\mathbf{x}^t+1\right)^q$$




$$\begin{aligned}K(\mathbf{x},\mathbf{y})&=\left(\mathbf{x}^T\mathbf{y}+1\right)^2\\&=\left(x_1y_1+x_2y_2+1\right)^2\\&=1+2x_1y_1+2x_2y_2+2x_1x_2y_1y_2+x_1^2y_1^2+x_2^2y_2^2\end{aligned}$$

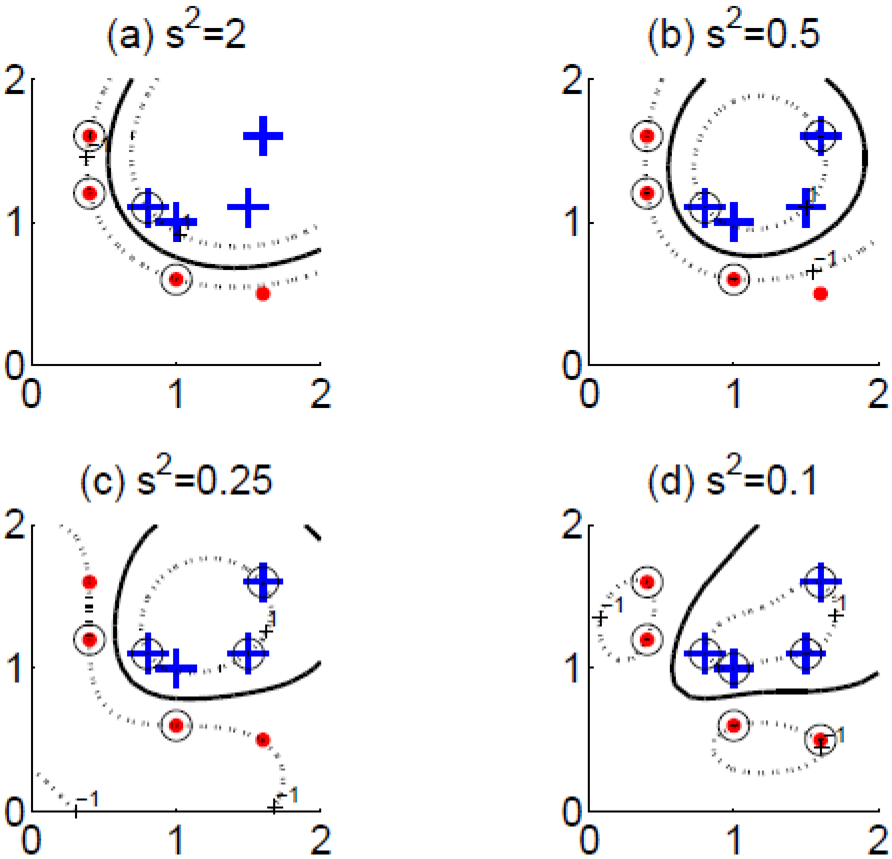
$$\phi(\mathbf{x})=\left[1,\sqrt{2}x_1,\sqrt{2}x_2,\sqrt{2}x_1x_2,x_1^2,x_2^2\right]^T$$

# Vectorial Kernels



- Radial-basis functions:

$$K(\mathbf{x}^t, \mathbf{x}) = \exp\left[-\frac{\|\mathbf{x}^t - \mathbf{x}\|^2}{2s^2}\right]$$



# Defining kernels



- Kernel "engineering"
- Defining good measures of similarity
- String kernels, graph kernels, image kernels, ...
- Empirical kernel map: Define a set of templates $\boldsymbol{m}_i$ and score function $s(\mathbf{x},\boldsymbol{m}_i)$

  $\phi(\mathbf{x}^t)=[s(\mathbf{x}^t,\boldsymbol{m}_1), s(\mathbf{x}^t,\boldsymbol{m}_2),\ldots, s(\mathbf{x}^t,\boldsymbol{m}_M)]$

  and

  $K(\mathbf{x},\mathbf{x}^t)=\phi(\mathbf{x})^T \phi(\mathbf{x}^t)$

# Multiple Kernel Learning



- Fixed kernel combination

$$K(\mathbf{x},\mathbf{y}) = \begin{cases} cK(\mathbf{x},\mathbf{y}) \\ K_1(\mathbf{x},\mathbf{y}) + K_2(\mathbf{x},\mathbf{y}) \\ K_1(\mathbf{x},\mathbf{y})K_2(\mathbf{x},\mathbf{y}) \end{cases}$$

- Adaptive kernel combination

$$K(\mathbf{x},\mathbf{y}) = \sum_{i=1}^{m} \eta_i K_i(\mathbf{x},\mathbf{y})$$

$$L_d = \sum_t \alpha^t - \frac{1}{2}\sum_t \sum_s \alpha^t \alpha^s r^t r^s \sum_i \eta_i K_i(\mathbf{x}^t, \mathbf{x}^s)$$

$$g(\mathbf{x}) = \sum_t \alpha^t r^t \sum_i \eta_i K_i(\mathbf{x}^t, \mathbf{x})$$

- Localized kernel combination $g(\mathbf{x}) = \sum_t \alpha^t r^t \sum_i \eta_i(\mathbf{x} \mid \theta) K_i(\mathbf{x}^t, \mathbf{x})$

# Multiclass Kernel Machines



- 1-vs-all
- Pairwise separation
- Error-Correcting Output Codes (section 17.5)
- Single multiclass optimization

$$\min \frac{1}{2}\sum_{i=1}^{K}\|\mathbf{w}_i\|^2 + C\sum_i\sum_t \xi_i^t$$

subject to

$$\mathbf{w}_{z^t}{}^T\mathbf{x}^t + w_{z^t0} \geq \mathbf{w}_i{}^T\mathbf{x}^t + w_{i0} + 2 - \xi_i^t, \forall i \neq z^t, \xi_i^t \geq 0$$

# SVM for Regression



- Use a linear model (possibly kernelized)

$$f(\mathbf{x}) = \mathbf{w}^T\mathbf{x} + w_0$$

- Use the $\epsilon$-sensitive error function

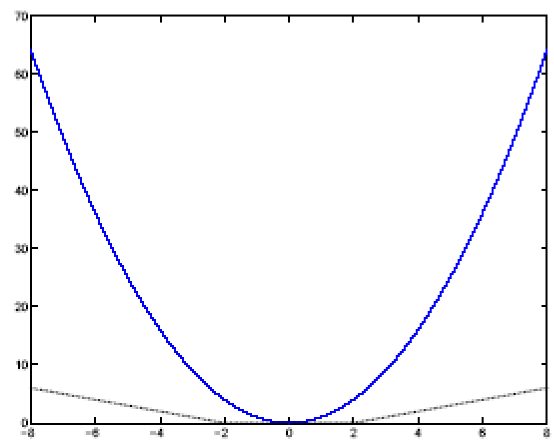
$$e_\varepsilon\left(r^t, f\left(\mathbf{x}^t\right)\right) = \begin{cases} 0 & \text{if } \left|r^t - f\left(\mathbf{x}^t\right)\right| < \varepsilon \\ \left|r^t - f\left(\mathbf{x}^t\right)\right| - \varepsilon & \text{otherwise} \end{cases}$$

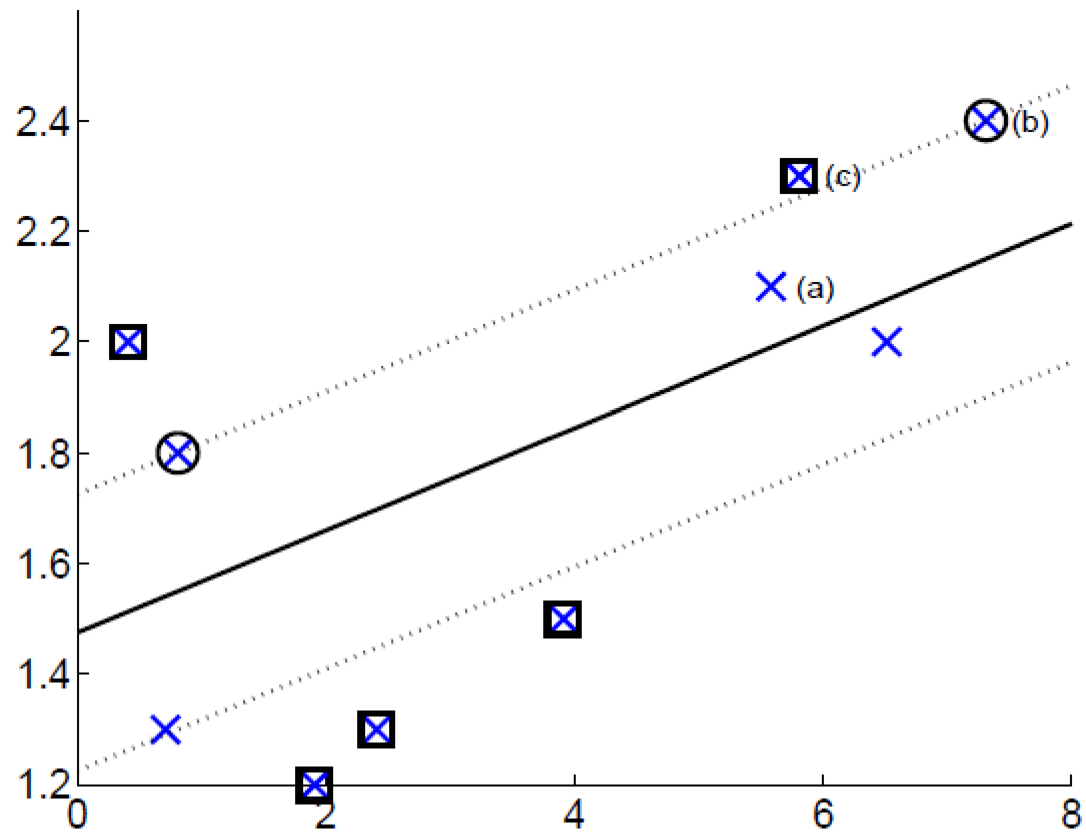
$$\min \frac{1}{2}\|\mathbf{w}\|^2 + C\sum_t \left(\xi_+^t + \xi_-^t\right)$$

$$r^t - \left(\mathbf{w}^T\mathbf{x} + w_0\right) \le \varepsilon + \xi_+^t$$

$$\left(\mathbf{w}^T\mathbf{x} + w_0\right) - r^t \le \varepsilon + \xi_-^t$$

$$\xi_+^t, \xi_-^t \ge 0$$

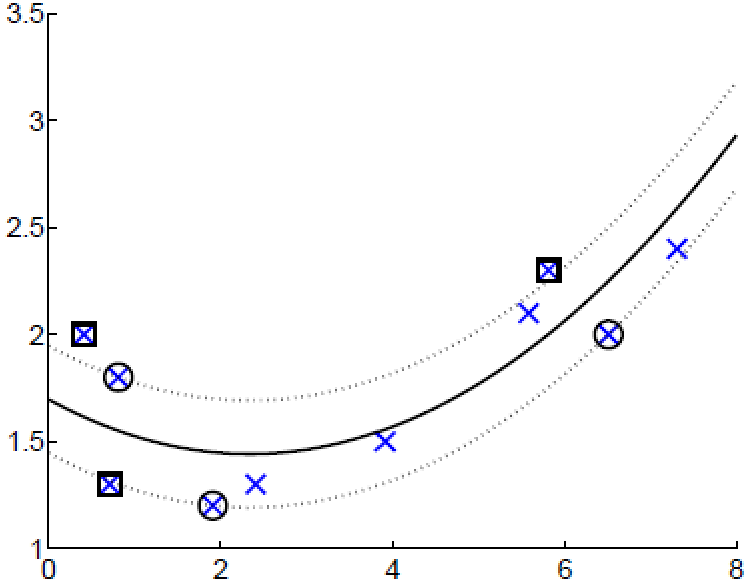


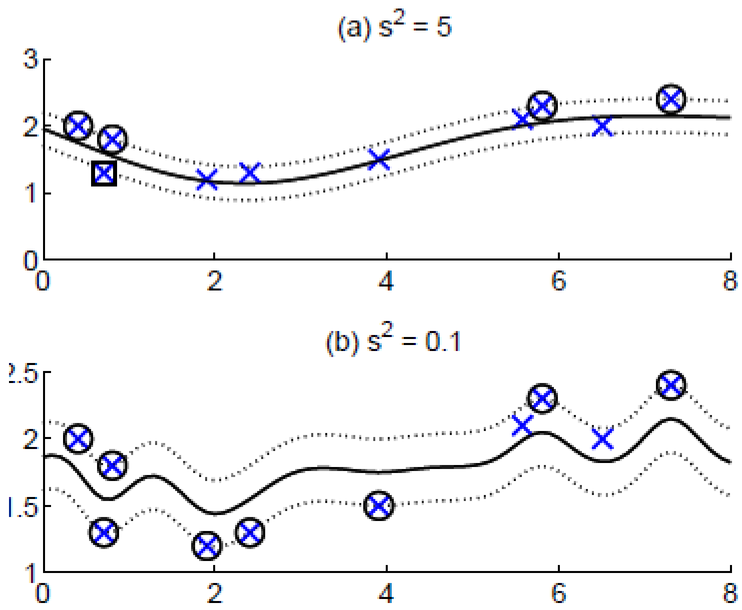

2.4
2.2
2
1.8
1.6
1.4
1.2
0
2
4
6
8
(a)
(b)
(c)

# Kernel Regression



- Polynomial kernel
- Gaussian kernel



# Kernel Machines for Ranking



- We require not only that scores be correct order but at least +1 unit margin.
- Linear case:

$$\min \frac{1}{2}\|\mathbf{w}_i\|^2 + C\sum_t \xi_i^t$$

subject to

$$\mathbf{w}^T\mathbf{x}^u \geq \mathbf{w}^T\mathbf{x}^v + 1 - \xi^t, \forall t : r^u \prec r^v, \xi_i^t \geq 0$$

# One-Class Kernel Machines



- Consider a sphere with center ***a*** and radius *R*

$$\min R^2 + C\sum_t \xi^t$$

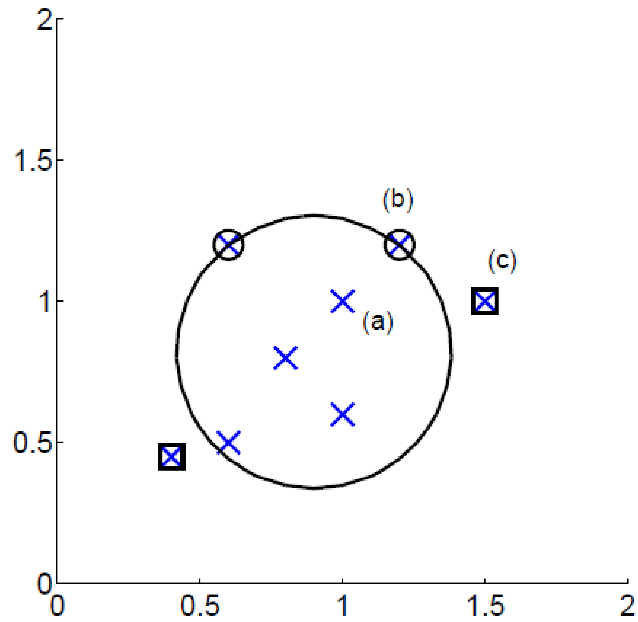
subject to

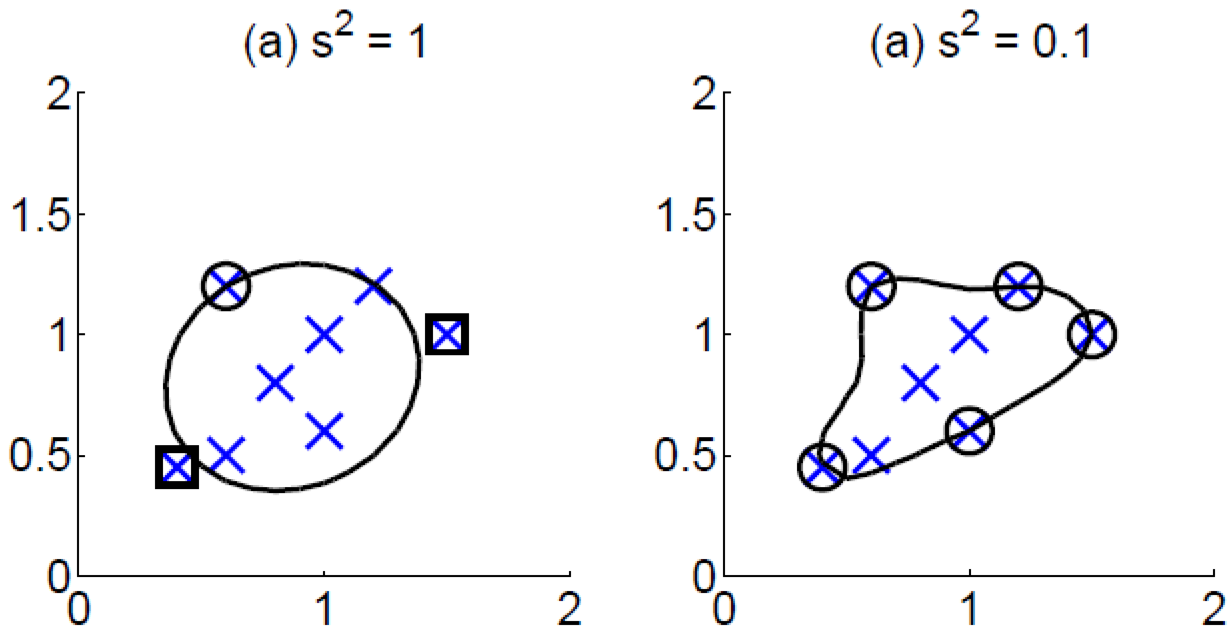
$$\left\|\mathbf{x}^t - a\right\| \le R^2 + \xi^t, \xi^t \ge 0$$

$$L_d = \sum_t \alpha^t \left(x^t\right)^T x^s - \sum_{t=1}^{N}\sum_s \alpha^t \alpha^s r^t r^s \left(x^t\right)^T x^s$$

subject to

$$0 \le \alpha^t \le C, \sum_t \alpha^t = 1$$





(a) s2 = 1
(a) s2 = 0.1
2
1.5
1
0.5
0
0
1
2

# Large Margin Nearest Neighbor



- Learns the matrix **M** of Mahalanobis metric

  $D(\mathbf{x}^i, \mathbf{x}^j)=(\mathbf{x}^i-\mathbf{x}^j)^T\mathbf{M}(\mathbf{x}^i-\mathbf{x}^j)$

- For three instances *i*, *j*, and *l*, where *i* and *j* are of the same class and *l* different, we require

  $D(\mathbf{x}^i, \mathbf{x}^l) > D(\mathbf{x}^i, \mathbf{x}^j)+1$

  and if this is not satisfied, we have a slack for the difference and we learn M to minimize the sum of such slacks over all *i*,*j*,*l* triples (*j* and *l* being one of *k* neighbors of *i*, over all *i*)

# Learning a Distance Measure



- LMNN algorithm (Weinberger and Saul 2009)

$$(1-\mu)\sum_{i,j}\mathcal{D}(\mathbf{x}^i,\mathbf{x}^j)+\mu\sum_{i,j,l}(1-y_{il})\xi_{ijl}$$

subject to

$$\begin{aligned}\mathcal{D}(\mathbf{x}^i,\mathbf{x}^l) &\geq \mathcal{D}(\mathbf{x}^i,\mathbf{x}^j)+1-\xi^{ijl},\ \text{if } \boldsymbol{r}^i=\boldsymbol{r}^j \text{ and } \boldsymbol{r}^i\neq\boldsymbol{r}^l\\ \xi^{ijl} &\geq 0\end{aligned}$$

- LMCA algorithm (Torresani and Lee 2007) uses a similar approach where **M**=**L**$^T$**L** and learns **L**

# Kernel Dimensionality Reduction



- Kernel PCA does PCA on the kernel matrix (equal to canonical PCA with a linear kernel)
- Kernel LDA, CCA