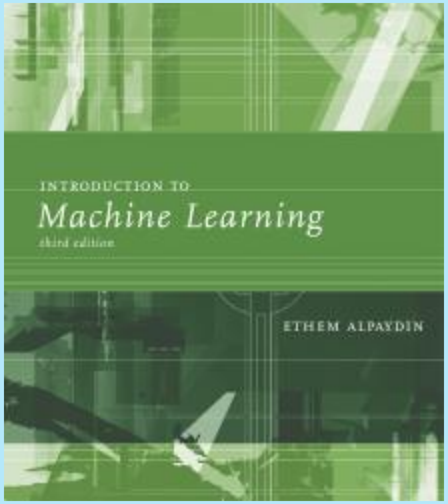


Lecture Slides for

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

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

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

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

Note that we do not simply require

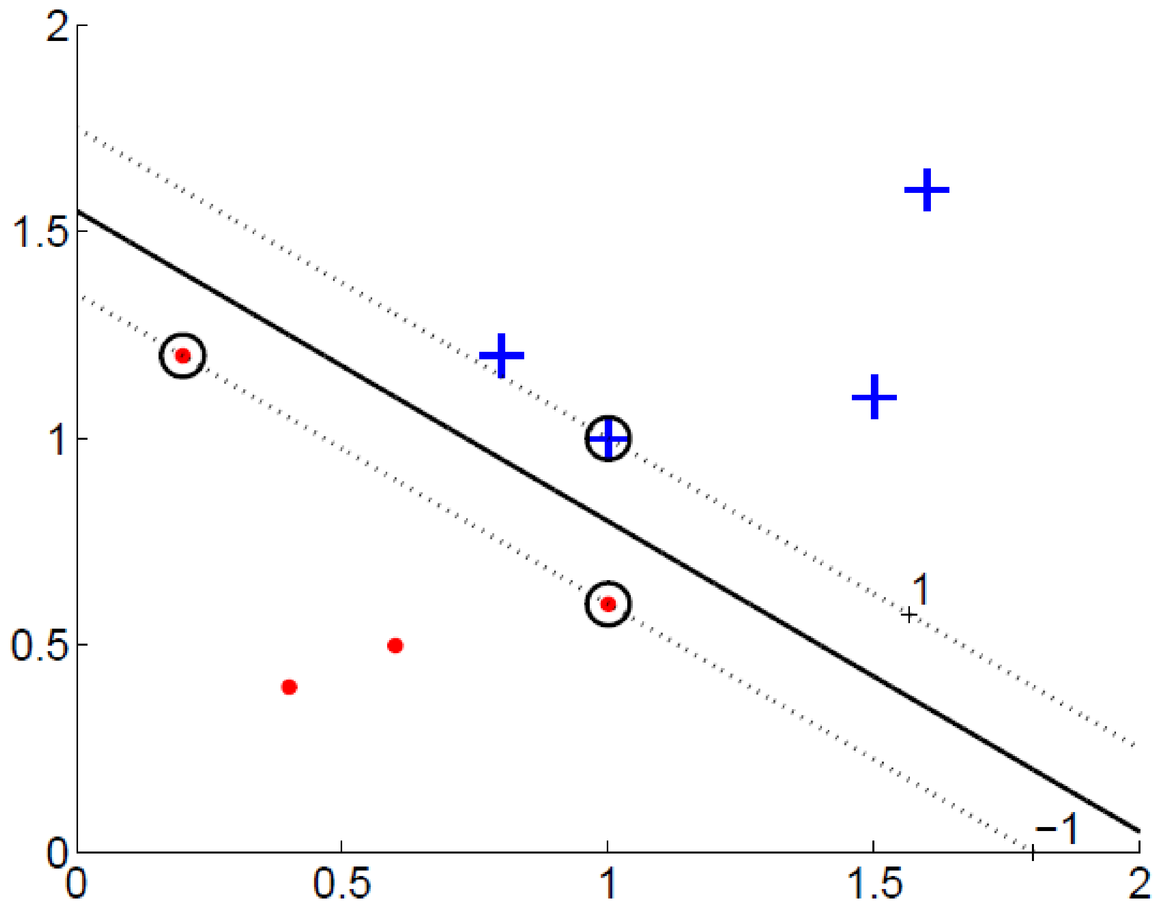
$$r^t \left(\mathbf{w}^T \mathbf{x}^t + w_0 \right) \geq 0$$

Margin

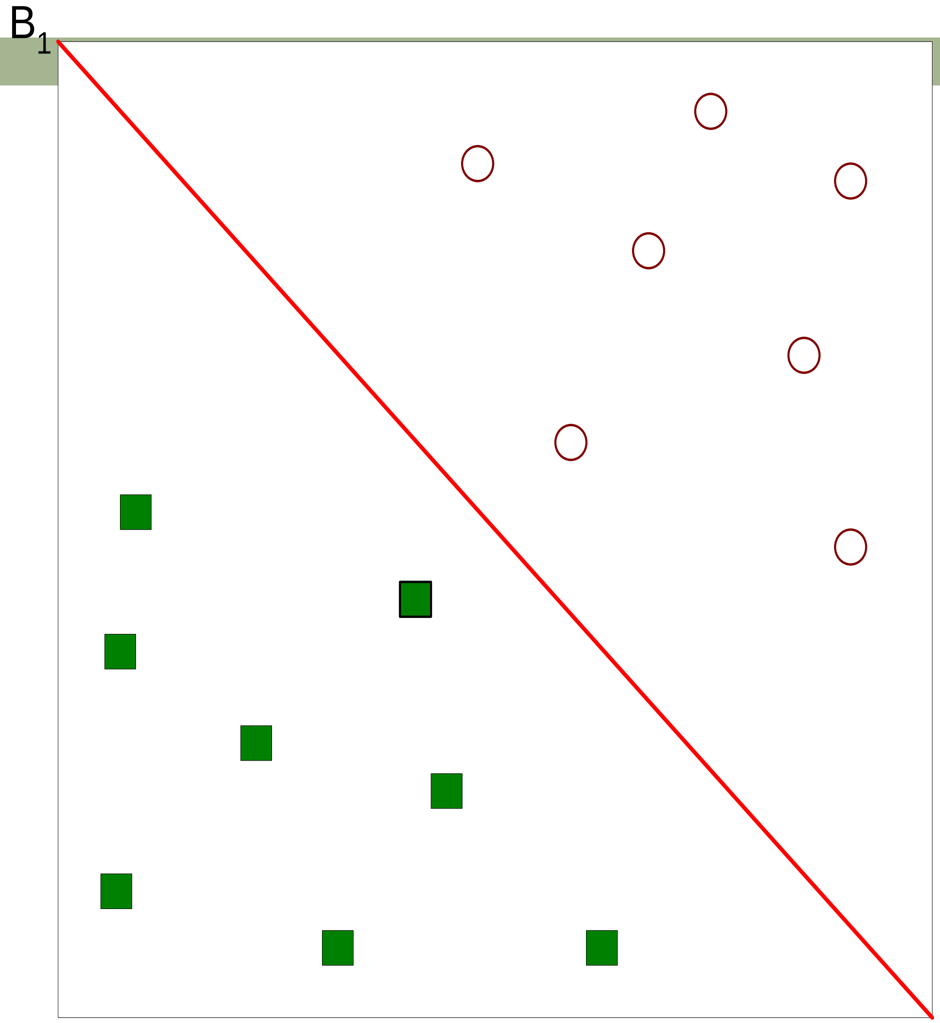
- Distance from the discriminant to the closest instances on either side
- Distance of **x** to the hyperplane is $$\frac{\left|\mathbf{w}^T\mathbf{x}^t + w_0\right|}{\|\mathbf{w}\|}$$
- We require $$\frac{r^t\left(\mathbf{w}^T\mathbf{x}^t + w_0\right)}{\|\mathbf{w}\|} \geq \rho, \forall t$$
- For a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

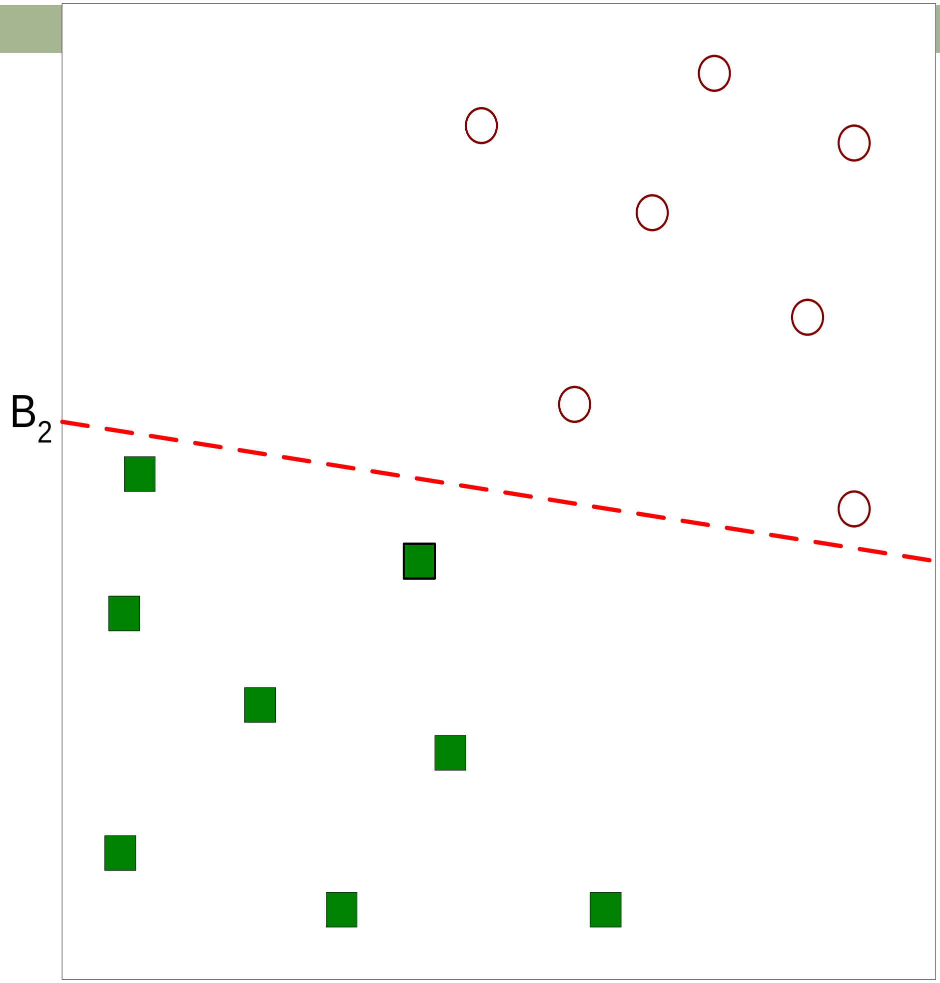


Support Vector Machines



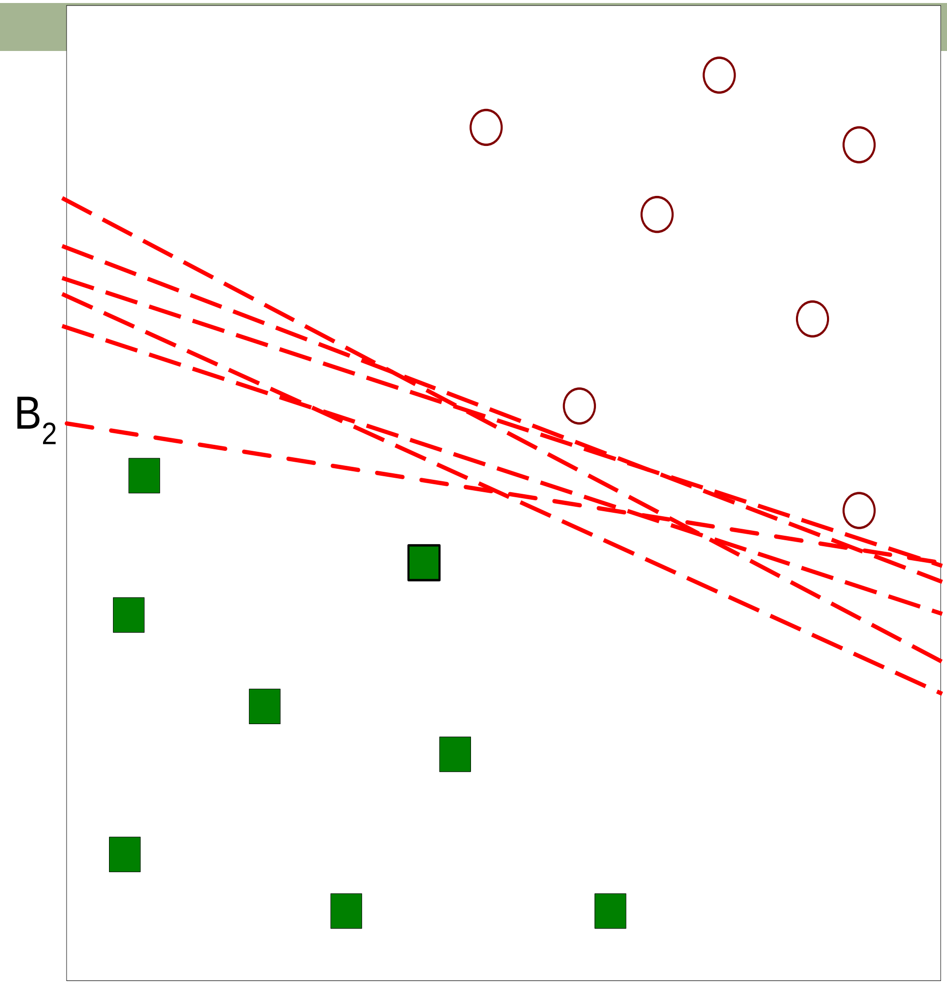
- SVMs use a single hyperplane; one Possible Solution

Support Vector Machines



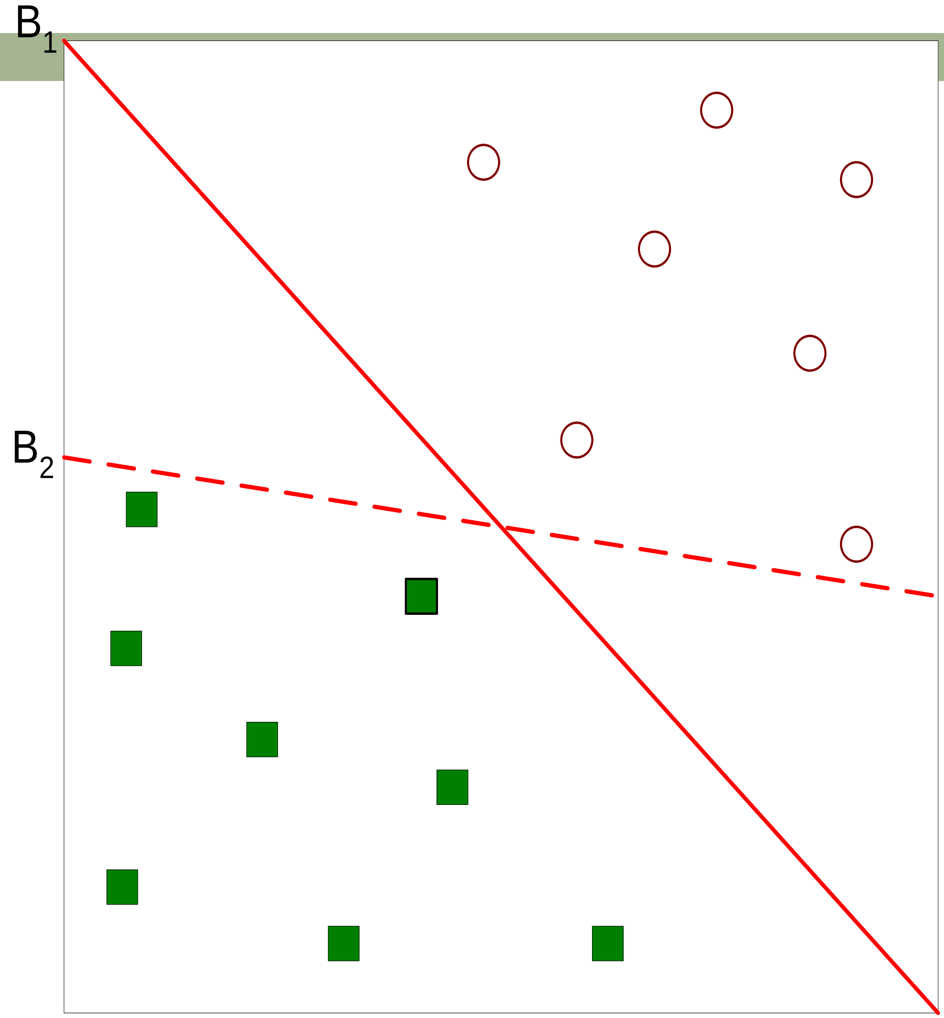
- Another possible solution

Support Vector Machines



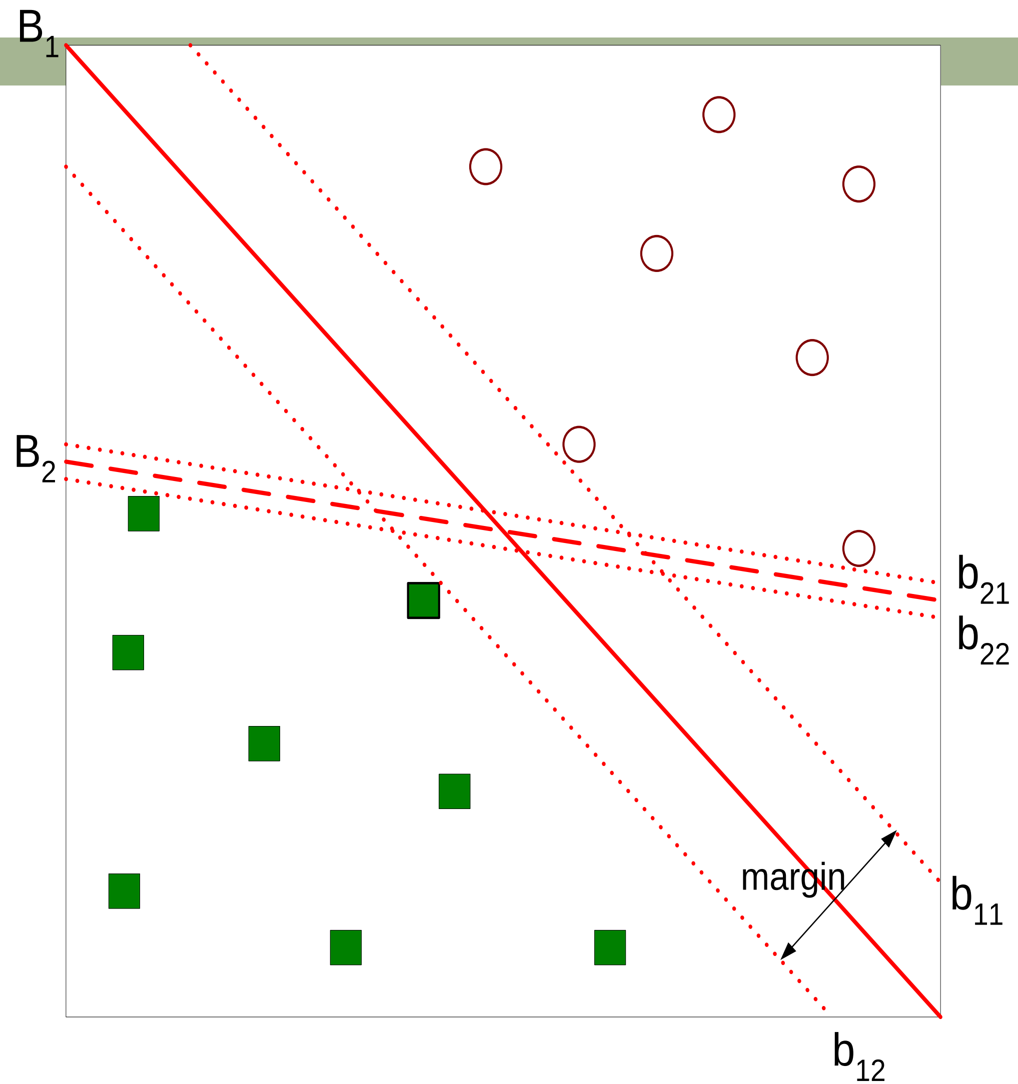
- Other possible solutions

Support Vector Machines



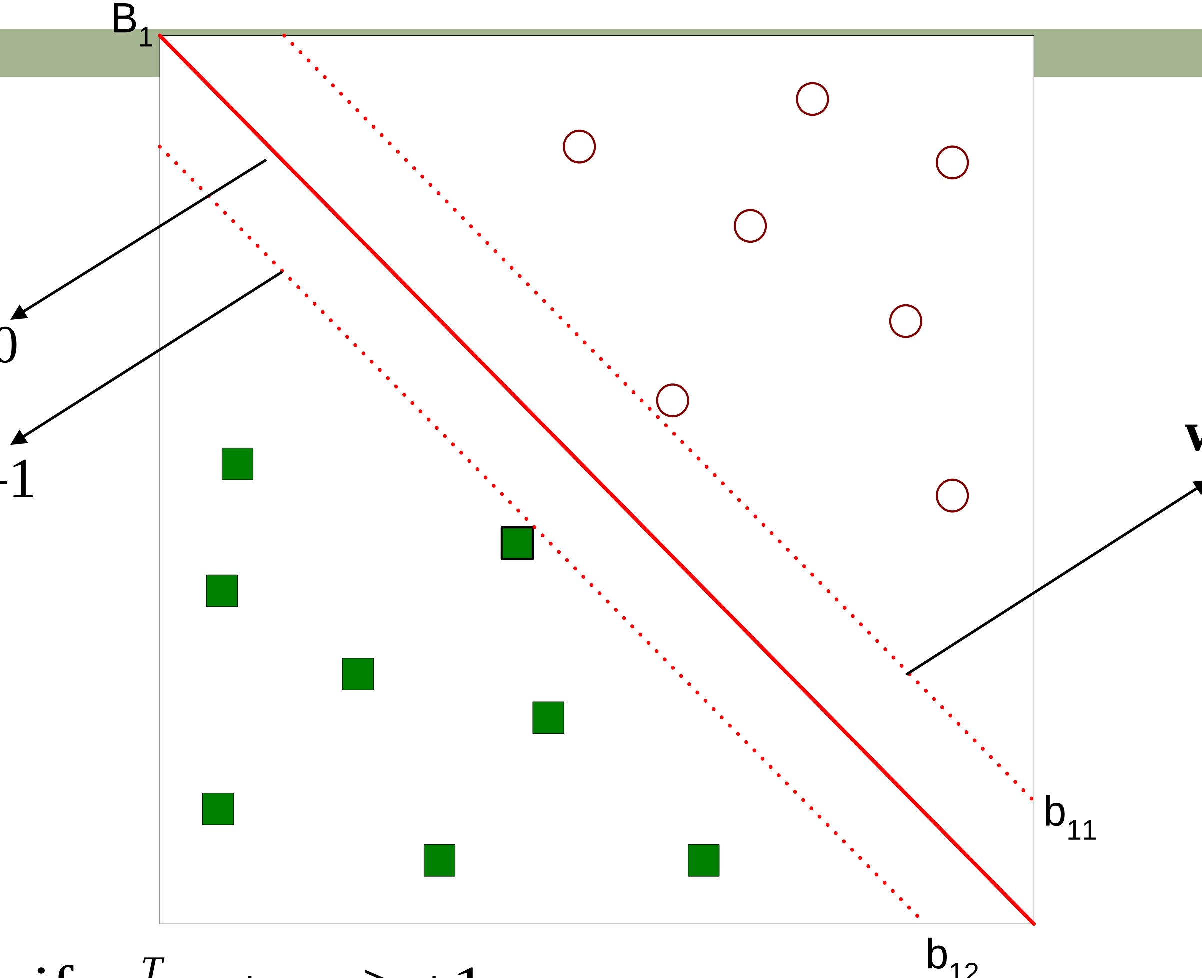
- Which one is better? B_1 or B_2?
- How do you define better?

Support Vector Machines



- Find a hyperplane maximizing the margin => B_1 is better than B_2

Support Vector Machines



$$\mathbf{w}^T.\mathbf{x} + w_0 = 0$$

$$\mathbf{w}^T.\mathbf{x} + w_0 = -1$$

$$\mathbf{w}^T.\mathbf{x} + w_0 = +1$$

$$f(\mathbf{x}) = \begin{cases} 1 & \text{if } \mathbf{w}^T.\mathbf{x} + w_0 \geq +1 \\ -1 & \text{if } \mathbf{w}^T.\mathbf{x} + w_0 \leq -1 \end{cases}$$

$$\text{Margin} = \frac{2}{\|\mathbf{w}\|}$$

- To max margin

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

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

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

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

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

$$\alpha^t \geq 0$$

To maximize the dual with respect to α^t only

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

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

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

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

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

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

Linear SVM for Non-linearly Separable Problems

No kernel

Measures prediction error

- What if the problem is not linearly separable?
  - Introduce slack variables
  - Need to minimize:

Parameter

$$L(\mathbf{w}) = \frac{\|\mathbf{w}\|^2}{2} + C\left(\sum_{t=1}^{N} \xi^t\right)$$

Inverse size of margin between hyperplanes

Slack variable

allows constraint violation to a certain degree

  - C is chosen using a validation set trying to keep the margins wide while keeping the training error low.

Soft Margin Hyperplane

- Not linearly separable

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

متغیرهای سستی، تسامح، مسامحه *slack variables*

- We define Soft error as: $\sum_t \xi^t$

The number of misclassifications is $\#\{\xi^t > 1\}$

- New primal is

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

where μ^t are the new Lagrange parameters to guarantee the positivity of ξ^t .

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

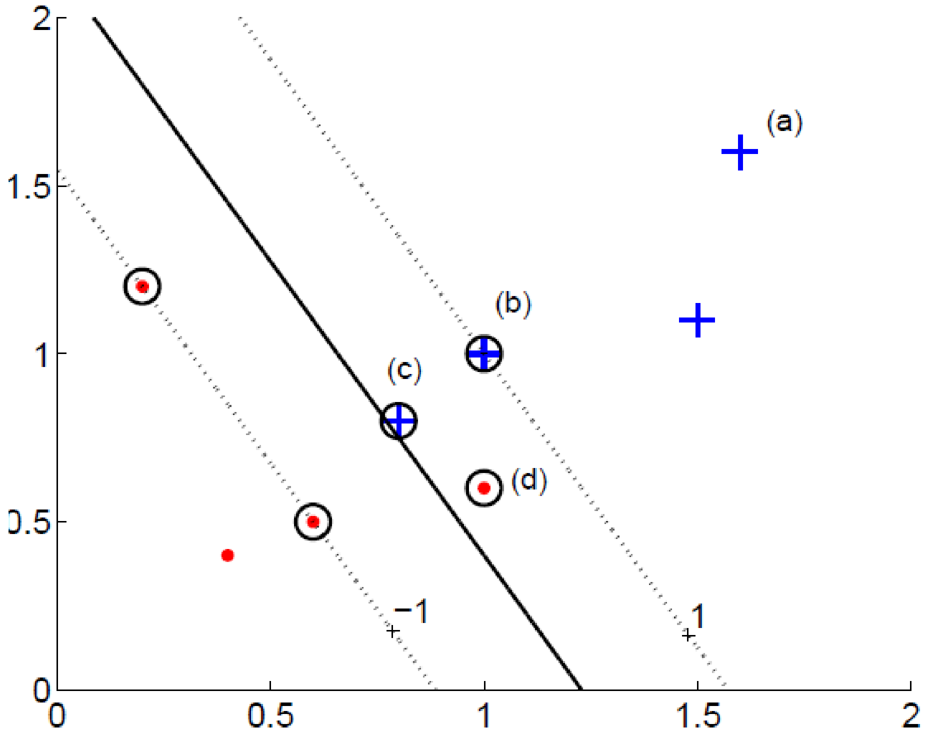
$$\frac{\partial L_p}{\partial w_0} = \sum_{t=1}^{N} \alpha^t r^t = 0, \quad \frac{\partial L_p}{\partial \xi^t} = C - \alpha^t - \mu^t = 0$$

$$\mu^t \geq 0 \Rightarrow 0 \leq \alpha^t \leq C \Rightarrow$$

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

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

$$E_N[P(error)] \leq \frac{E_N[\text{\# of support vectors}]}{N}$$

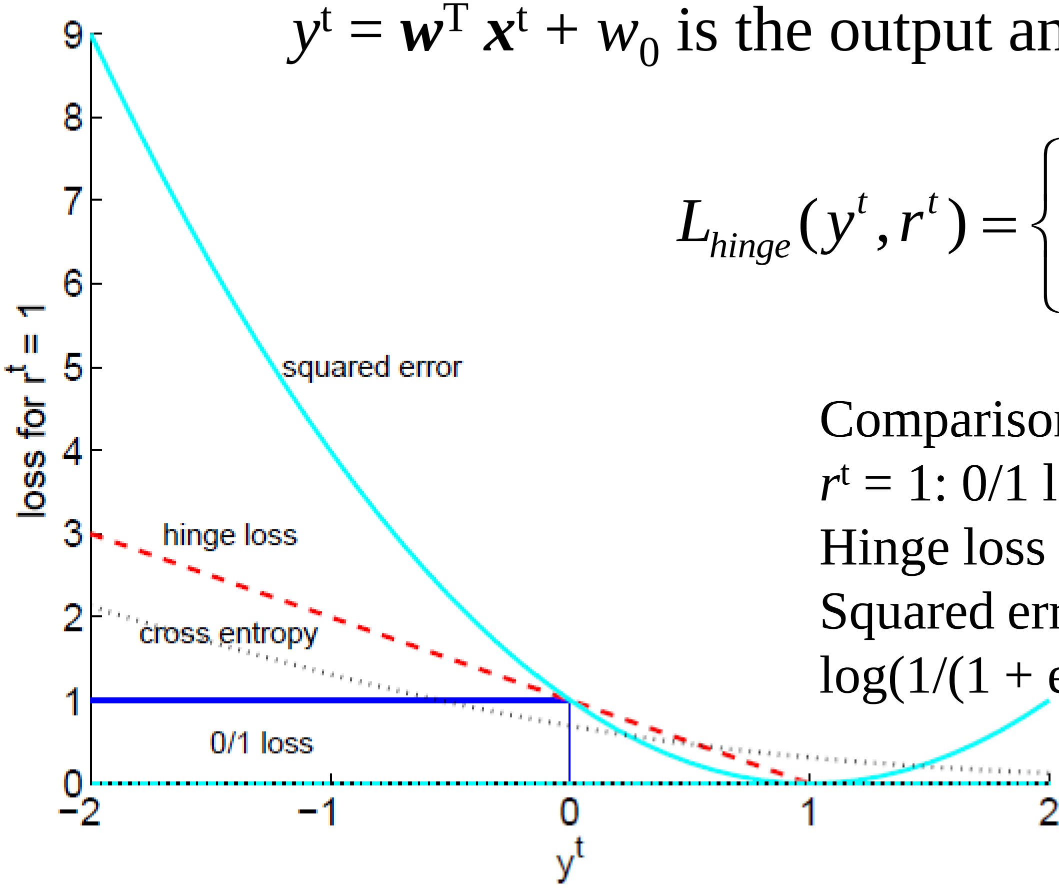


In classifying an instance, there are four possible cases: In (a), the instance is on the correct side and far away from the margin; $r^t g(\boldsymbol{x}^t)>1$, $\xi^t = 0$. In (b), $\xi^t = 0$; it is on the right side and on the margin. In (c), $\xi^t = 1 - g(\boldsymbol{x}^t)$, $0 < \xi < 1$; it is on the right side but is in the margin and not sufficiently away. In (d), $\xi^t = 1 + g(\boldsymbol{x}^t) > 1$; it is on the wrong side—this is a misclassification. All cases except (a) are support vectors. In terms of the dual variable, in (a), $\alpha^t = 0$; in (b), $\alpha^t < C$; in (c) and (d), $\alpha^t = C$.

*Hinge Loss

We define error if the instance is on the wrong side or if the margin is less than 1. This is called the *hinge loss*. If $y^t = \boldsymbol{w}^T \boldsymbol{x}^t + w_0$ is the output and r^t is the desired output:

$$L_{hinge}(y^t, r^t) = \begin{cases} 0 & \text{if } y^t r^t \geq 1 \\ 1 - y^t r^t & \text{otherwise} \end{cases}$$



Comparison of different loss functions for $r^t = 1$: 0/1 loss is 0 if $y^t = 1$, 1 otherwise. Hinge loss is 0 if $y^t > 1$, $1 - y^t$ otherwise. Squared error is $(1 - y^t)^2$. Cross-entropy is $\log(1/(1 + \exp(-y^t)))$.

* ν-SVM

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

subject to

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

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

subject to

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

ν controls the fraction of support vectors

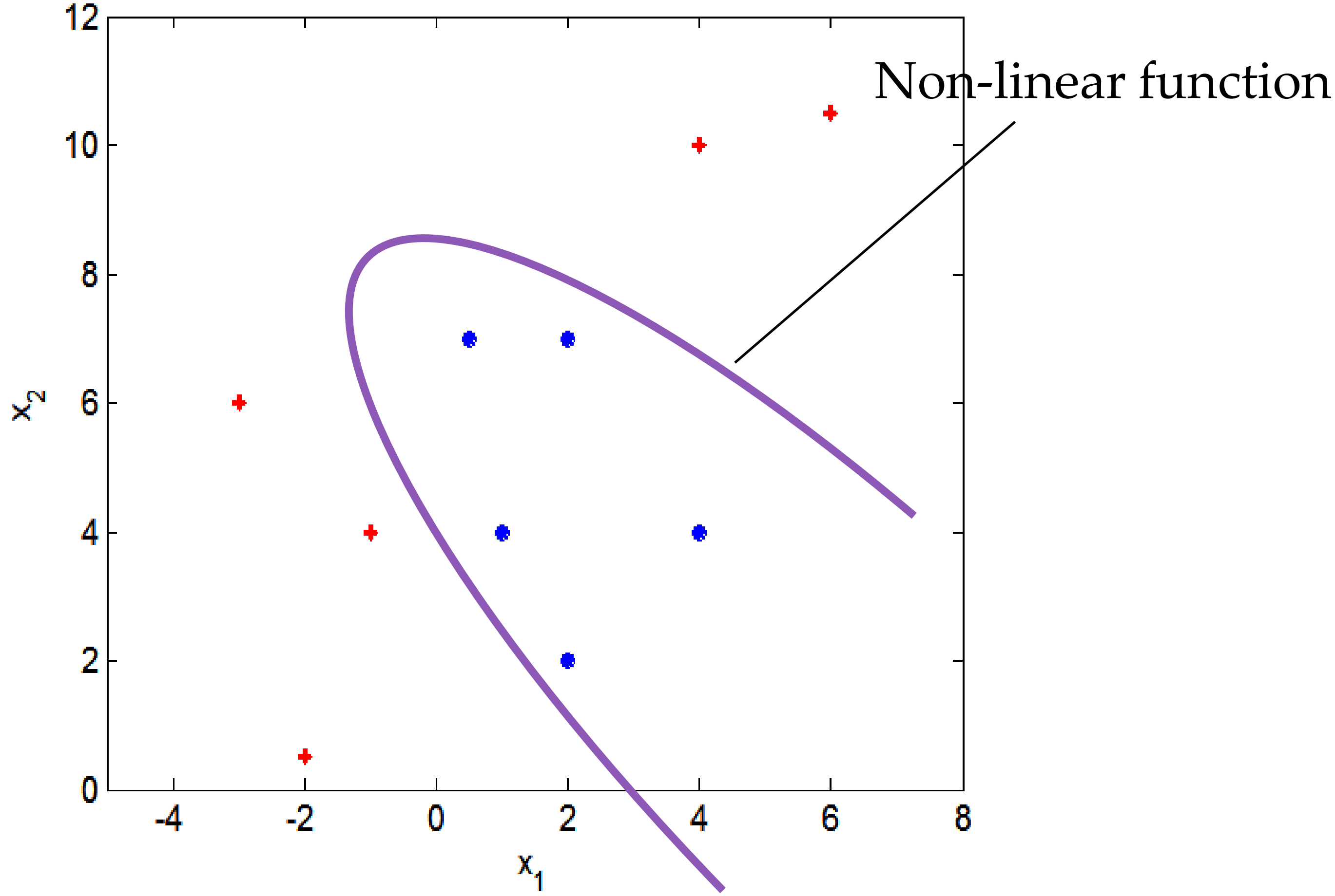
Key Properties of Support Vector Machines

1. Use a single hyperplane which subdivides the space into two half-spaces, one which is occupied by Class1 and the other by Class2.
2. They maximize the margin of the decision boundary using quadratic optimization techniques which find the optimal hyperplane.
3. When used in practice, SVM approaches frequently map (using ϕ) the examples to a higher dimensional space and find margin maximal hyperplanes in the mapped space, obtaining decision boundaries which are not hyperplanes in the original space.
4. Moreover, versions of SVMs exist that can be used when linear separability cannot be accomplished.

Nonlinear Support Vector Machines

- What if decision boundary is not linear?

Alternative 1:
Use technique that Employs non-linear decision boundaries

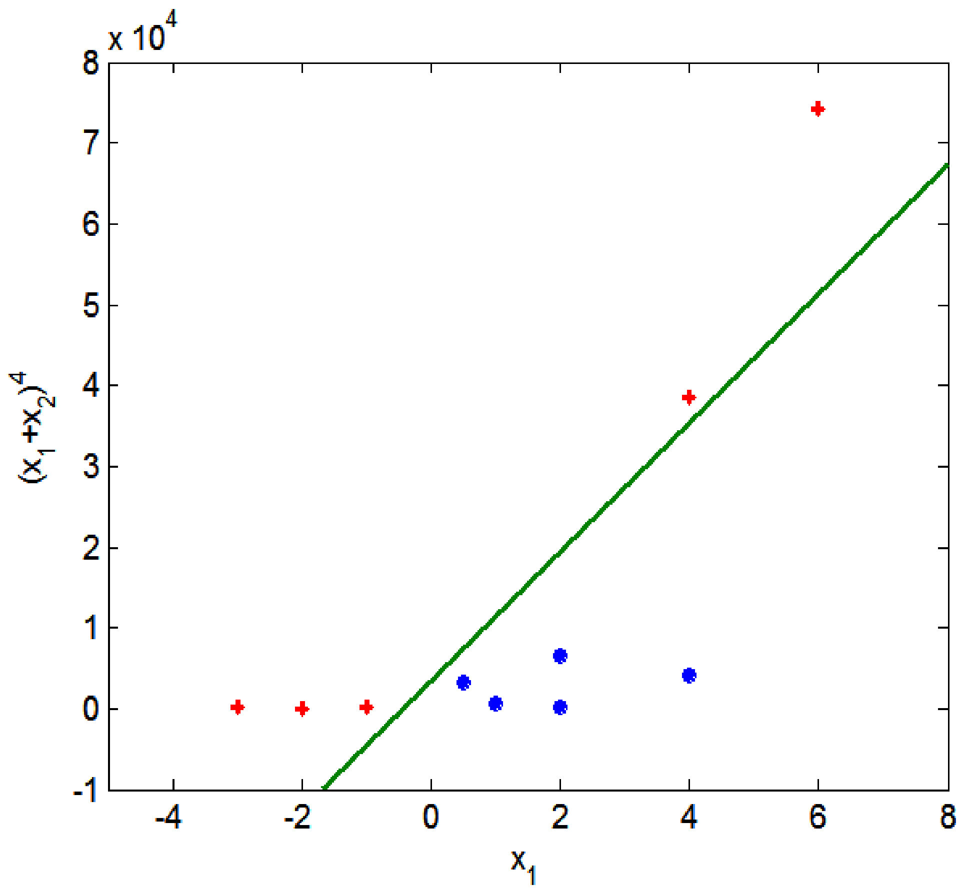


Nonlinear Support Vector Machines

1. Transform data into higher dimensional space
2. Find the best hyperplane using the methods introduced earlier

Alternative 2:

Transform into a higher dimensional attribute space and find linear decision boundaries in this space



Kernel Trick

- Preprocess input $\boldsymbol{x}$ by basis functions

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

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

- The SVM solution

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

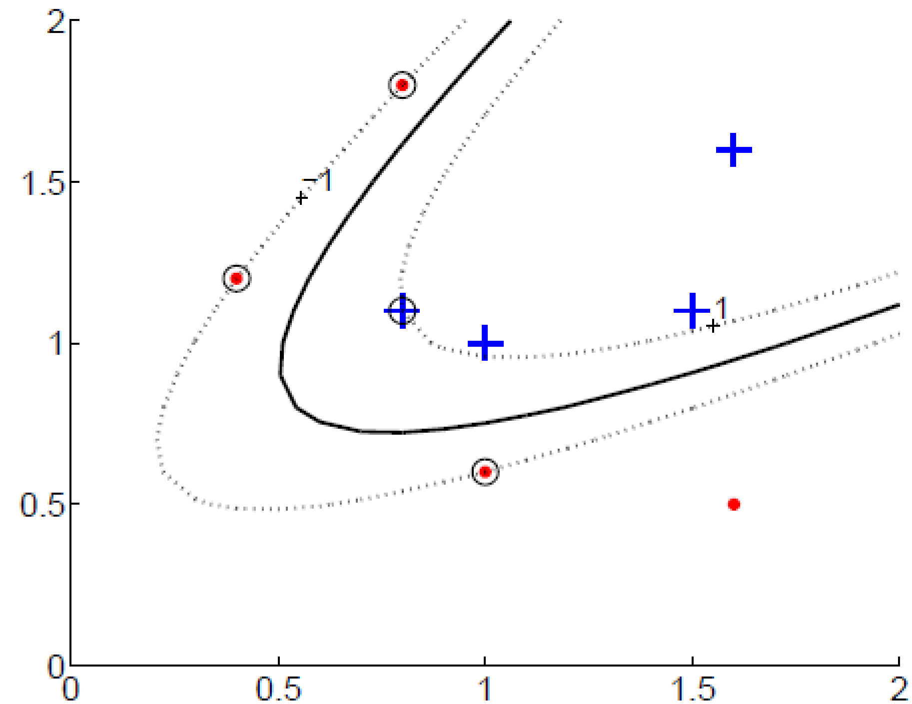
$$g\left(\mathbf{x}\right) = \mathbf{w}^T \boldsymbol{\varphi}\left(\mathbf{x}\right) = \sum_t \alpha^t r^t \boxed{\boldsymbol{\varphi}\left(\mathbf{x}^t\right)^T \boldsymbol{\varphi}\left(\mathbf{x}\right)}$$

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

Vectorial Kernels

- Polynomials of degree q:

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



$$K(\mathbf{x}, \mathbf{y}) = \left(\mathbf{x}^T \mathbf{y} + 1\right)^2$$

$$= \left(x_1 y_1 + x_2 y_2 + 1\right)^2$$

$$= 1 + 2x_1 y_1 + 2x_2 y_2 + 2x_1 x_2 y_1 y_2 + x_1^2 y_1^2 + x_2^2 y_2^2$$

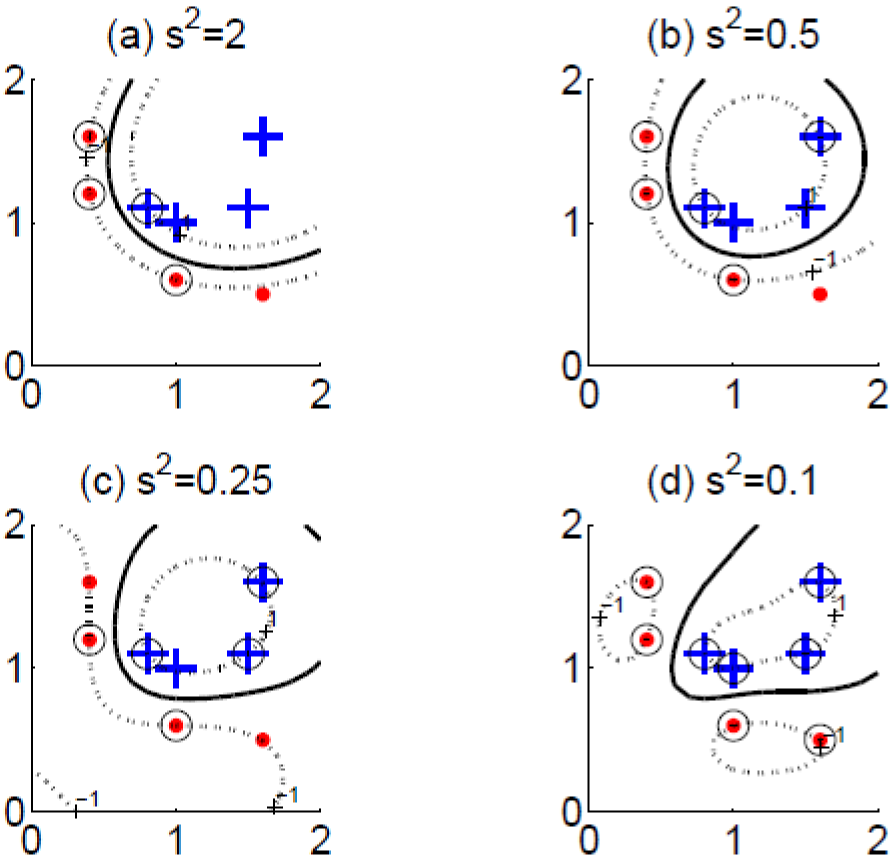
corresponds to the inner product of the basis function

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

Vectorial Kernels

- Radial-basis functions:

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



Defining kernels

- Kernels are generally considered to be measures of similarity.
- Kernel “engineering”.
- Defining good measures of similarity.
- String kernels, graph kernels, image kernels, ...
- Given two documents say D_1 and D_2, one possible representation is called bag of words where we predefine M words relevant for the application $\rightarrow$ $\varphi(D_1)^T \varphi(D_2)$ counts the number of shared words.

Defining kernels

- Given two strings (of genes), a kernel measures the edit distance, namely, how many operations (insertions, deletions, substitutions) it takes to convert one string into another; this is also called alignment.
- Empirical kernel map: Define a set of templates $\boldsymbol{m}_i$ and score function $s(\boldsymbol{x},\boldsymbol{m}_i)$

  $$\varphi(\boldsymbol{x}^t)=[s(\boldsymbol{x}^t,\boldsymbol{m}_1), s(\boldsymbol{x}^t,\boldsymbol{m}_2),..., s(\boldsymbol{x}^t,\boldsymbol{m}_M)]^{\mathrm{T}}$$

  and we define the empirical kernel map as

  $$K(\boldsymbol{x}^t,\boldsymbol{x}^s)=\varphi(\boldsymbol{x}^t)^T \varphi(\boldsymbol{x}^s)$$

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 (*K(K−1)/2* classifiers)
- Error-Correcting Output Codes (section 17.5)
- Single multiclass optimization

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

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

- The one-vs.-all approach is generally preferred because it solves K separate N variable problems whereas the multiclass formulation uses $K \cdot N$ variables.

* SVM for Regression

- Use a linear model (possibly kernelized)

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

- Use the ε-sensitive error function

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

- which means that we tolerate errors up to ε and also that errors beyond have a linear effect and not a quadratic one. This error function is therefore more tolerant to noise and is thus more robust.

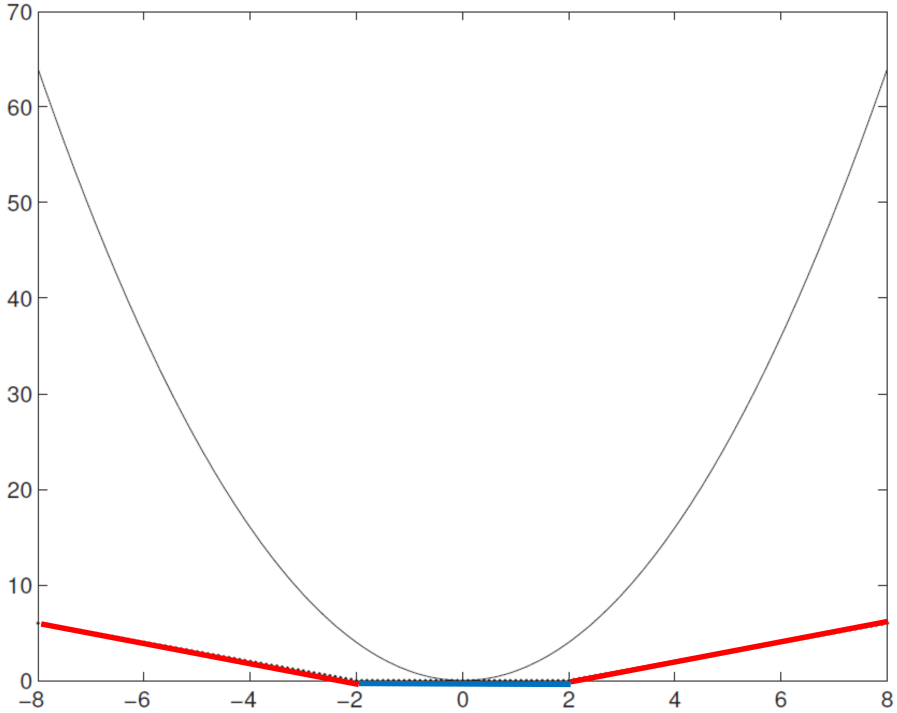
$$\min \frac{1}{2}\|\mathbf{w}\|^2 + C\sum_t \left(\xi_+^t + \xi_-^t\right) \quad \text{subject to}$$

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

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

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

we use two types of slack variables, for positive and negative deviations, to keep them positive.



The dual is

$$
\begin{aligned}
L_d \quad = \quad & -\frac{1}{2}\sum_t \sum_s (\alpha_+^t - \alpha_-^t)(\alpha_+^s - \alpha_-^s)(\boldsymbol{x}^t)^T \boldsymbol{x}^s \\
& -\epsilon \sum_t (\alpha_+^t + \alpha_-^t) + \sum_t r^t (\alpha_+^t - \alpha_-^t)
\end{aligned}
$$

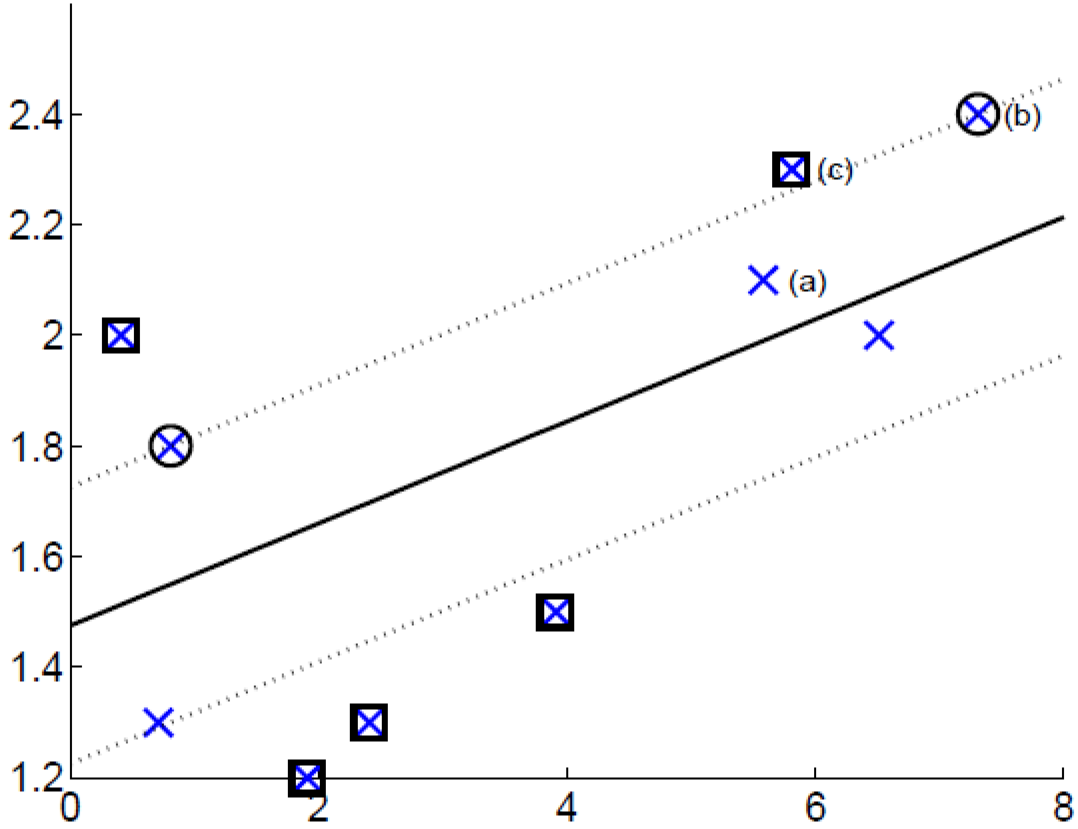
subject to $\quad 0 \le \alpha_+^t \le C \,,\, 0 \le \alpha_-^t \le C \,,\, \sum_t (\alpha_+^t - \alpha_-^t) = 0$

Once we solve this, we see that all instances that fall in the tube have $\alpha^t{}_+ = \alpha^t{}_- = 0$; these are the instances that are fitted with enough precision. The support vectors satisfy either $\alpha^t{}_+ > 0$ or $\alpha^t{}_- > 0$ and are of two types. They may be instances that are on the boundary of the tube (either $\alpha^t{}_+$ or $\alpha^t{}_-$ is between 0 and C), and we use these to calculate w_0 .

we can write the fitted line as a weighted sum of the support vectors:

$$
f(\mathbf{x}) = \left(\mathbf{w}^T\mathbf{x} + w_0\right) = \sum_t (\alpha_+^t - \alpha_-^t)(\mathbf{x}^t)^T\mathbf{x} + w_0
$$

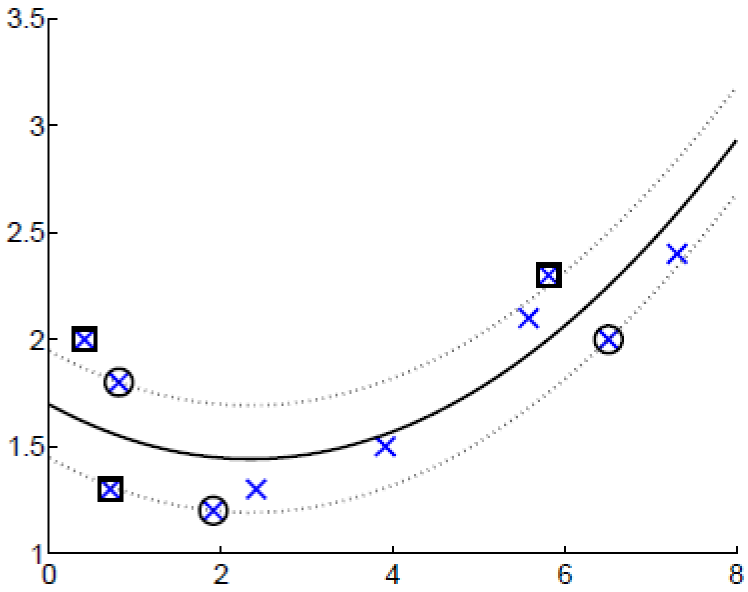
Note: $(\boldsymbol{x}^t)^T \boldsymbol{x}$ be replaced with Kernel $K(\boldsymbol{x}^t, \boldsymbol{x})$.



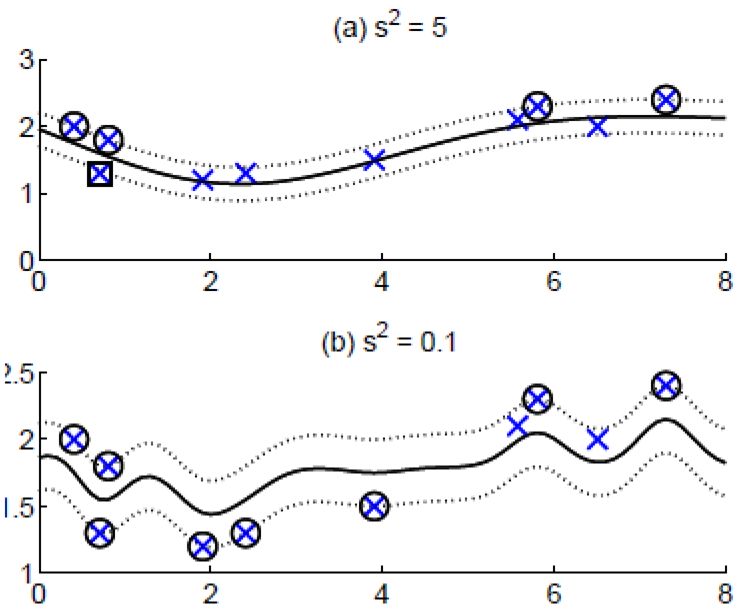
The fitted regression line to data points shown as crosses and the ε- tube are shown ($C = 10$, $\varepsilon = 0.25$). There are three cases: In (a), the instance is in the tube; in (b), the instance is on the boundary of the tube (circled instances); in (c), it is outside the tube with a positive slack, that is, $\xi^t_+ > 0$ (squared instances). (b) and (c) are support vectors. In terms of the dual variable, in (a), $\alpha^t_+ = 0, \alpha^t_- = 0$, in (b), $\alpha^t_+ < C$, and in (c), $\alpha^t_+ = C$.

* 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:

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

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

The dual is
$$L_d = \sum_t \alpha^t - \sum_t \sum_s \alpha^t \alpha^s (\mathbf{x}^u - \mathbf{x}^v)^T (\mathbf{x}^k - \mathbf{x}^l)$$

$$\text{subject to} \quad 0 \leq \alpha^t \leq C,$$

For new test instance ***x***, the score is calculated as $g(\mathbf{x}) = \sum_t \alpha^t \left(\mathbf{x}^u - \mathbf{x}^v\right)^T \mathbf{x}$

* One-Class Kernel Machines

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

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

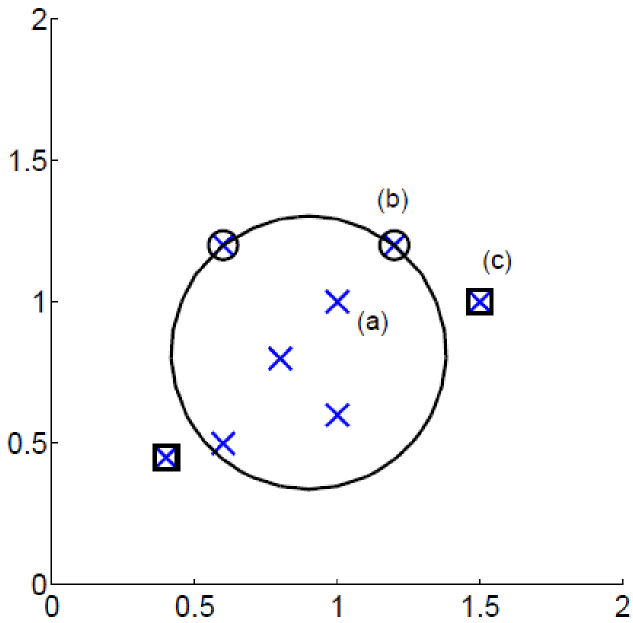
subject to

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

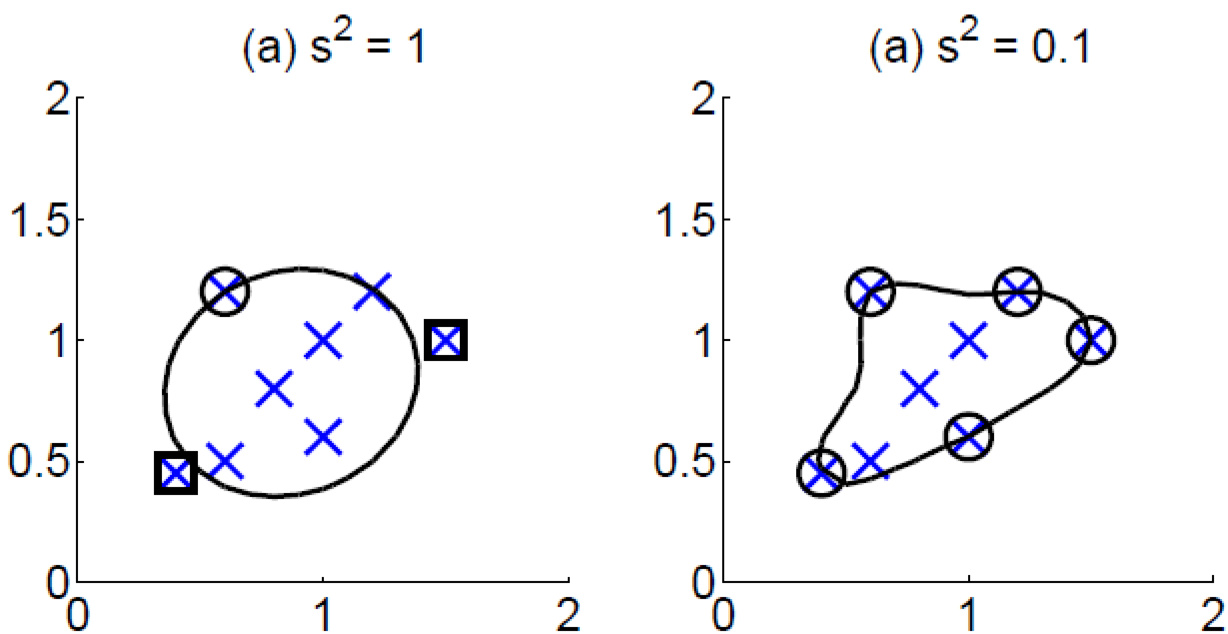
$$L_d = \sum_t \alpha^t \boxed{\left(x^t\right)^T x^s} - \sum_{t=1}^{N}\sum_s \alpha^t \alpha^s r^t r^s \boxed{\left(x^t\right)^T x^s}$$

subject to

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



One-class support vector machine places the smoothest boundary (here using a linear kernel, the circle with the smallest radius) that encloses as much of the instances as possible. There are three possible cases: In (a), the instance is a typical instance. In (b), the instance falls on the boundary with $\xi^t = 0$; such instances define R. In (c), the instance is an outlier with $\xi^t > 0$. (b) and (c) are support vectors. In terms of the dual variable, we have, in (a), $\alpha^t = 0$; in (b), $0 < \alpha^t < C$; in (c), $\alpha^t = C$.



One-class support vector machine using a Gaussian kernel with different spreads.

*Large Margin Nearest Neighbor

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

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

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

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

  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)

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

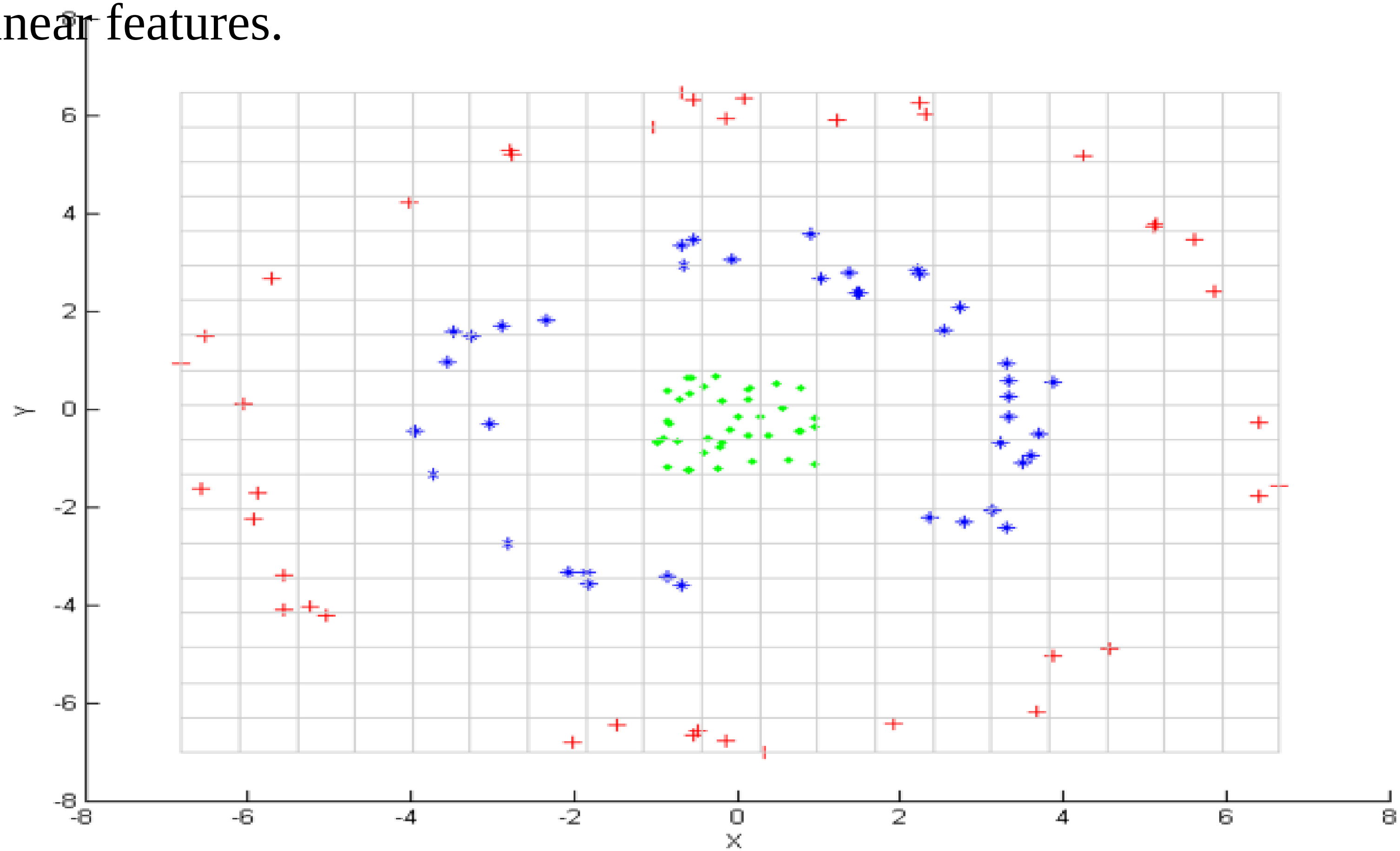
subject to

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

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

* Motivation Kernel PCA

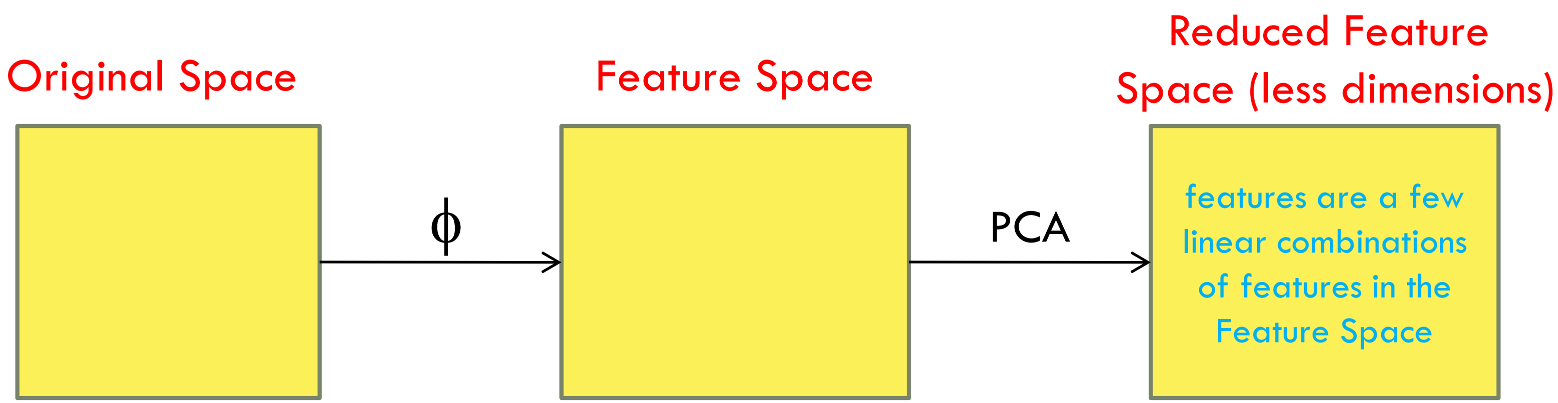
Example: we want to cluster the following dataset using K-means which will be difficult; **idea**: change coordinate system using a few new, non-linear features.



Remark: This approach uses kernels, but is unrelated to SVMs!

* Kernel PCA

- Kernel PCA does PCA on the kernel matrix (equal to doing PCA in the mapped space selecting some orthogonal eigenvectors in the mapped space as the new coordinate system)
- Kind of PCA using non-linear transformations in the original space, moreover, the vectors of the chosen new coordinate system are usually not orthogonal in the original space.
- Then, ML/DM algorithms are used in the Reduced Feature Space.



* Kernel Dimensionality Reduction

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