Introduction to Support Vector Machines (SVMs)

April 15, 2002

Outline

5. Nonlinear Support Vector Machines

6. References and Further Reading

1. Introduction

- originally due to Vladimir Vapnik (1960s)

- method for generating classification or regression functions from a set of labeled training data

- for a binary $\pm$ classification SVM, the basic idea is to find a hypersurface in the input space that splits the positive and negative examples in an optimal way (in the sense of better generalization)

- currently an active research area in the field of machine learning

2. Real-World Applications

There are several successful applications of SVMs to real-world problems:

- Text Categorization
- Handwriting Recognition
- Object Recognition
- Speaker Identification
- Face Detection

- Protein Homology Detection
- Gene Expression (Micro-Array Data Analysis)
- Protein Fold Recognition

3. Background Material

Consider a convex quadratic program:

$$\begin{aligned} & \min \ \tfrac{1}{2}\mathbf{x}^T Q\mathbf{x} + \mathbf{c}^T\mathbf{x} \\ \text{s.t.} \quad & \\ & A\mathbf{x} \geq \mathbf{b} \end{aligned} \tag{1}$$

- Q is a symmetric positive semidefinite $n \times n$ matrix

- $A \in R^{m\times n},\ \mathbf{c}, \mathbf{x} \in R^n,\ \mathbf{b} \in R^m$

Lagrangian:

$$L(\mathbf{x}, \lambda) = \frac{1}{2}\mathbf{x}^T Q\mathbf{x} + \mathbf{c}^T\mathbf{x} - \lambda^T(A\mathbf{x} - \mathbf{b}) \tag{2}$$

From optimization theory, we can solve (1) by finding a saddle point of L that minimizes L with respect to the *primal variable* $\mathbf{x}$ and maximizes L with respect to the *dual variable* λ.

Karush-Kuhn-Tucker (KKT) Conditions: Necessary and sufficient conditions for a point $\mathbf{x}^*$ to be a global minimizer of (1) is the existence of λ^* such that

$$\frac{\partial L(\mathbf{x}, \lambda)}{\partial \mathbf{x}}|_{\lambda=\lambda^*} = Q\mathbf{x}^* + \mathbf{c} - \lambda^{*T} A = 0 \tag{3}$$

$$\lambda^{*T}(A\mathbf{x}^* - \mathbf{b}) = 0 \tag{4}$$

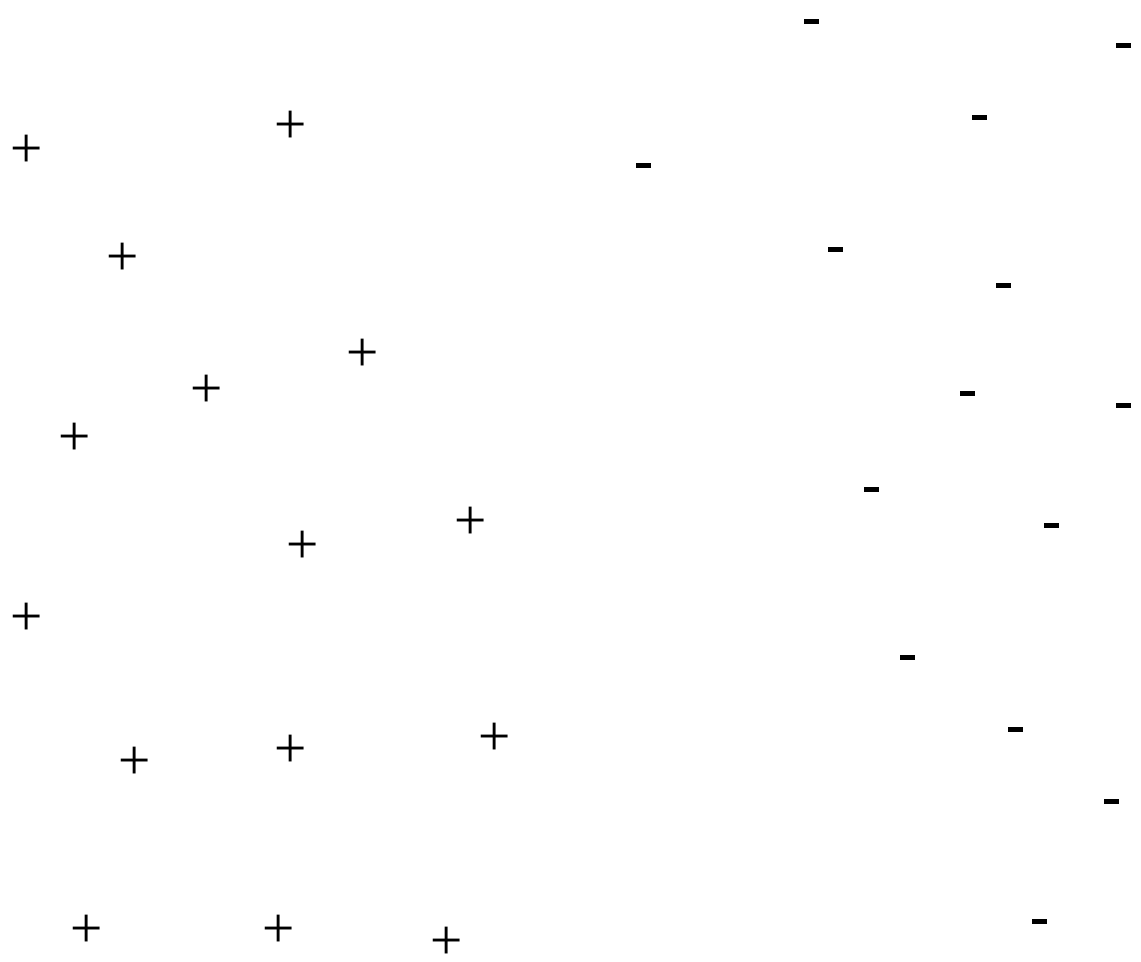
$$A\mathbf{x}^* \geq \mathbf{b} \tag{5}$$

$$\lambda^* \geq 0 \tag{6}$$

4. Linear Support Vector Machines

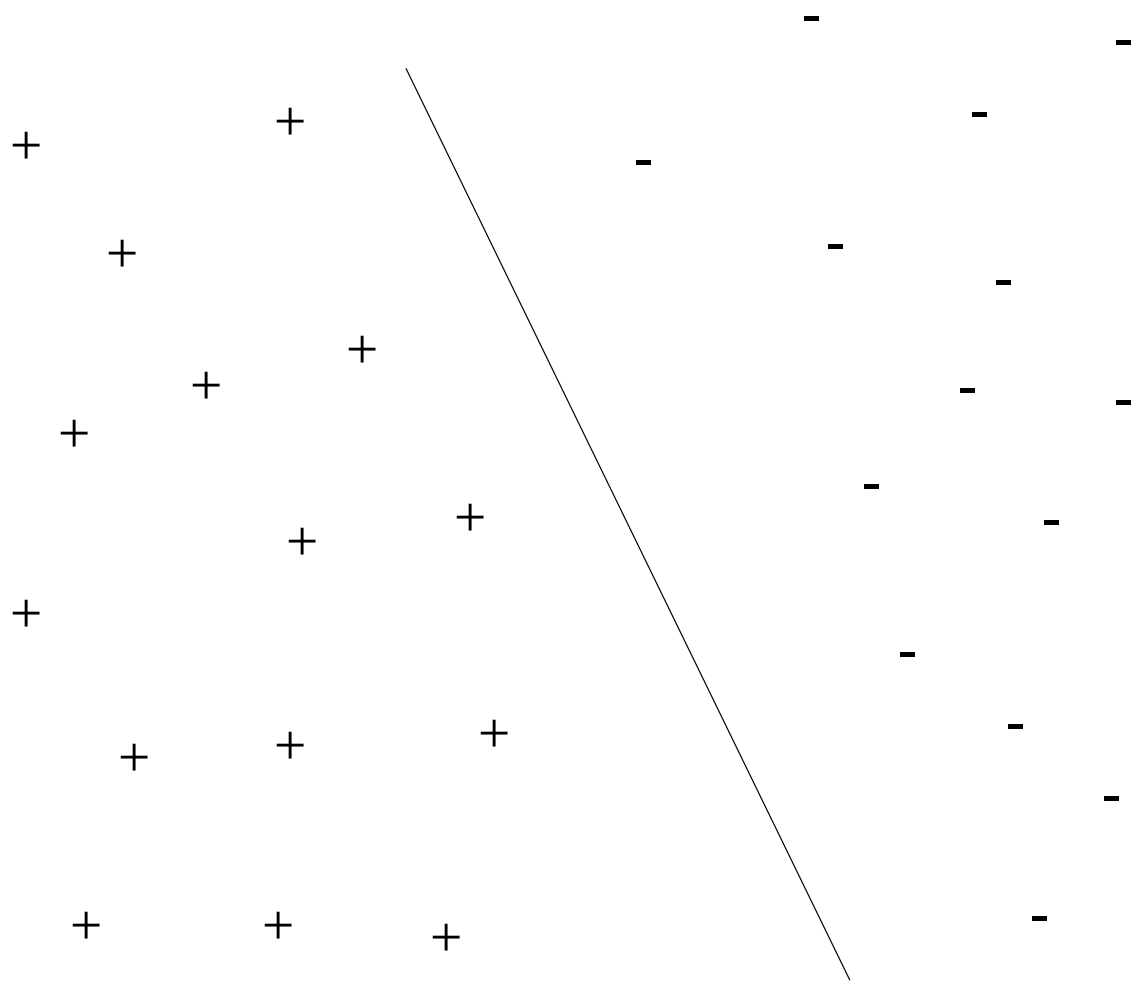
4.1 Linearly Separable Case

Consider a binary classification task with data points $\mathbf{x_1}, \ldots, \mathbf{x}_\ell \in R^d$ having corresponding labels $y_1, \ldots, y_\ell \in \{-1, +1\}$.



We shall refer to the set $\{(\mathbf{x}_1, y_1), \ldots, (\mathbf{x}_\ell, y_\ell)\} \subseteq R^d \times \{-1, +1\}$ as the set of *training examples*.

Suppose there exists a hyperplane which separates the positive and negative examples.



This means $\exists\ \mathbf{w} \in R^d, b \in R$ such that

$$\begin{cases} \mathbf{w}^T \mathbf{x}_i + b > 0 & \text{for } y_i = +1 \\ \mathbf{w}^T \mathbf{x}_i + b < 0 & \text{for } y_i = -1 \end{cases} \tag{7}$$

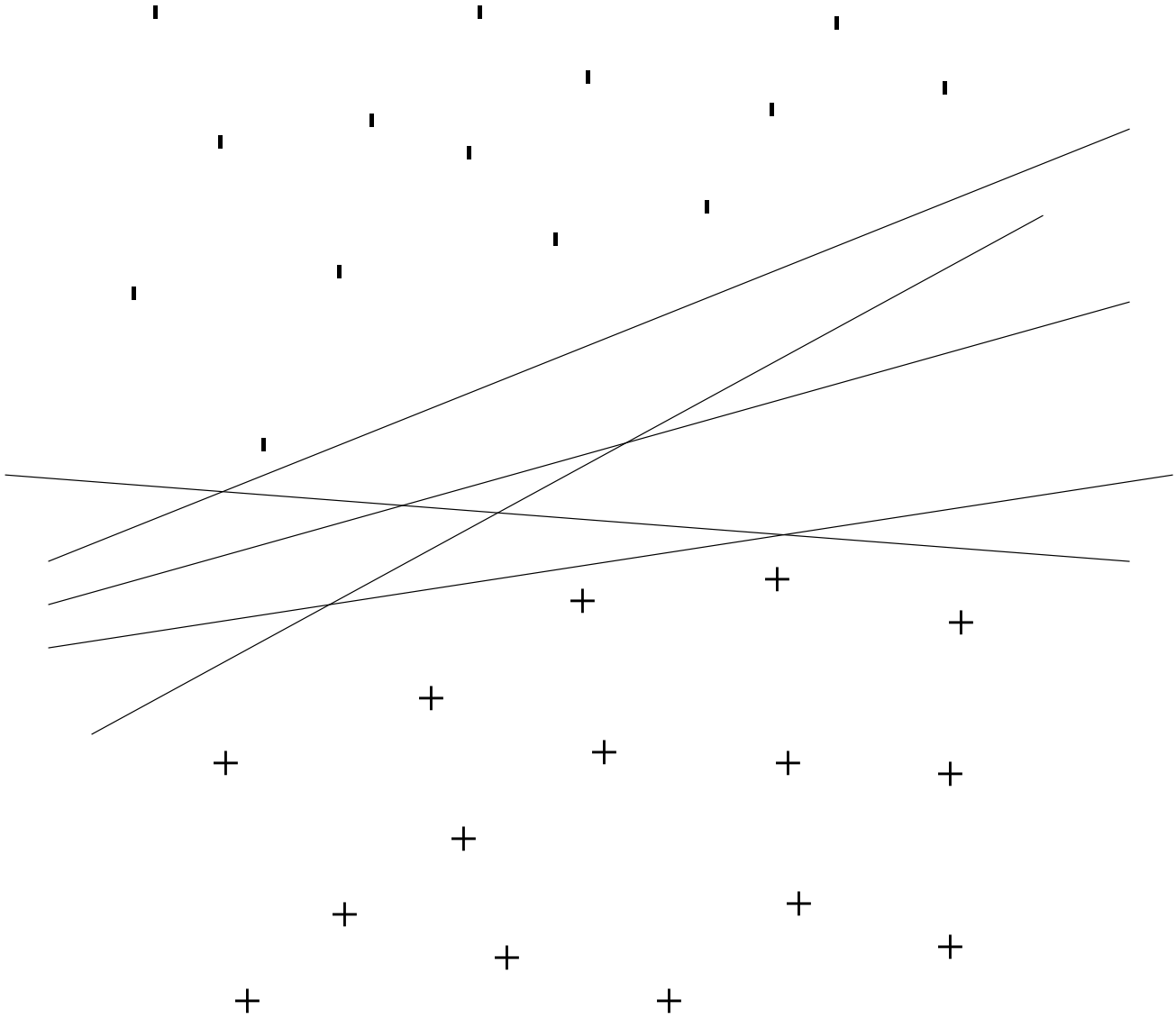
Note that this separating hyperplane may be used to define a decision function

$$f_{\mathbf{w},b}(\mathbf{x}) = \text{sgn}(\mathbf{w}^T \mathbf{x} + b), \quad \mathbf{x} \in R^d \tag{8}$$

Here, the *sign function* is defined as:

$$\text{sgn}(x) = \begin{cases} +1 & \text{if } x \geq 0 \\ -1 & \text{if } x < 0 \end{cases} \tag{9}$$

However, there is usually an infinite number of hyperplanes that can separate the positive and the negative examples.



Which one of these separating hyperplanes is the best? That is, which one will minimize the generalization error of the associated decision function?

According to the theory of structural risk minimization, the *optimal* hyperplane is the one yielding the maximum *margin of separation* between the two classes.

Here, we define the *margin* of a hyperplane $\mathbf{w}^T\mathbf{x} + b = 0$ as:

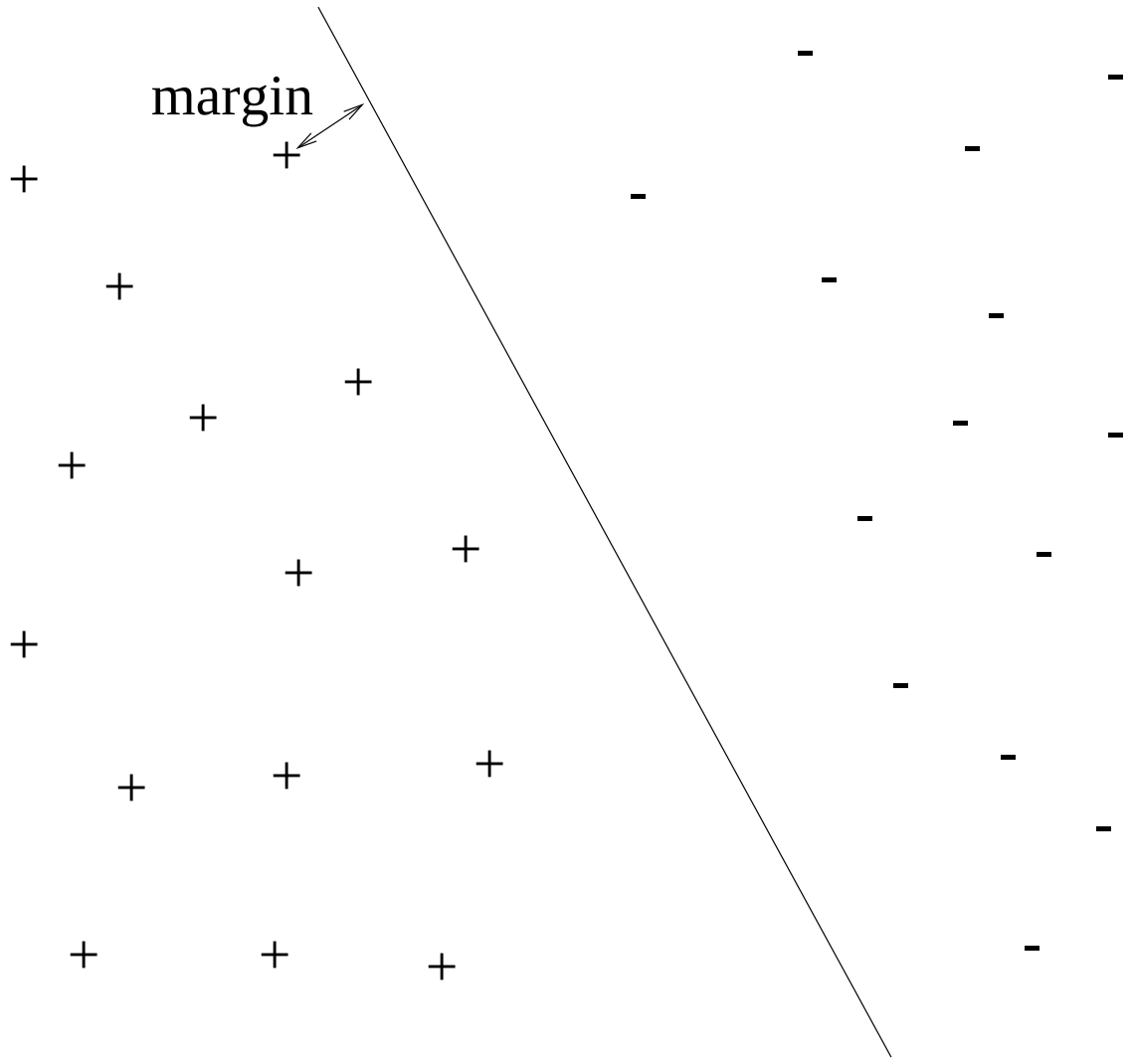
$$\min\{||\mathbf{x} - \mathbf{x}_i|| : \mathbf{x} \in R^d, \mathbf{w}^T\mathbf{x} + b = 0, i = 1, \ldots, \ell\} \quad (10)$$

That is, the margin of a separating hyperplane is the distance of the hyperplane to the closest data point.

Recall that the (perpendicular) distance between a point $\mathbf{x}_i$ and the hyperplane $\mathbf{w}^T\mathbf{x} + b = 0$ is given by

$$\frac{|\mathbf{w}^T\mathbf{x}_i + b|}{||\mathbf{w}||} \tag{11}$$

Hence, the margin is also the minimum of (11) over all i's.



How do we compute this optimal hyperplane?

Goal: Find $\mathbf{w} \in R^d$ and $b \in R$ that solves the following optimization problem:

$$\max_{\mathbf{w},b} \min_{i} \frac{|\mathbf{w}^T \mathbf{x}_i + b|}{||\mathbf{w}||} \quad \text{s.t.} \quad f_{\mathbf{w},b}(\mathbf{x}_i) = y_i \tag{12}$$

But, how do we solve this optimization problem?

First, observe that the decision function

$$f_{\mathbf{w},b}(x) = \text{sgn}(\mathbf{w}^T\mathbf{x} + b) \tag{13}$$

is exactly the same as

$$f_{c\mathbf{w},cb}(x) = \text{sgn}((c\mathbf{w})^T\mathbf{x} + cb) \tag{14}$$

for any $c > 0$.

Hence, it is possible to select a positive scaling of $\mathbf{w}$ and b so that

$$\min_{i=1,\ldots,\ell} |\mathbf{w}^T\mathbf{x}_i + b| = 1 \tag{15}$$

With requirement (15)

- the data point closest to the separating hyperplane has a distance of $1/||\mathbf{w}||$

- condition (7) is equivalent to

$$\begin{cases} \mathbf{w}^T\mathbf{x}_i + b \geq 1 & \text{for } y_i = +1 \\ \mathbf{w}^T\mathbf{x}_i + b \leq -1 & \text{for } y_i = -1 \end{cases} \tag{16}$$

Hence, the optimization problem is equivalent to the following quadratic program:

$$\begin{aligned} & \min_{\mathbf{w},b} \ \frac{1}{2}||\mathbf{w}||^2 \\ \text{s.t.} \quad & \\ & y_i(\mathbf{w}^T\mathbf{x}_i + b) \geq 1, \quad i = 1, \ldots, \ell \end{aligned} \tag{17}$$

The above problem can be solved by using standard quadratic programming (QP) techniques.

- can make the problem easier by considering a Lagrangian formulation of the problem

- this reformulation easily extends to nonlinear decision surfaces.

Hence, we construct the Lagrangian

$$L(\mathbf{w}, b, \alpha) = \frac{1}{2}||\mathbf{w}||^2 - \sum_{i=1}^{\ell} \alpha_i [y_i(\mathbf{w}^T \mathbf{x}_i + b) - 1], \qquad (18)$$

where $\alpha = (\alpha_1, \ldots, \alpha_\ell)^T$ is the vector of nonnegative Lagrange multipliers corresponding to the constraints in (17).

Now the original optimization problem can be solved by finding a saddle point of L which minimizes L with respect to the *primal variables* $\mathbf{w}_1, \ldots, \mathbf{w}_d$ and b, and maximizes L with respect to the *dual variables* $\alpha_1, \ldots, \alpha_\ell$.

Differentiating L with respect to the primal variables and setting the results equal to zero, we get:

$$\frac{\partial L(\mathbf{w}, b, \alpha)}{\partial \mathbf{w}} = w - \sum_{i=1}^{\ell} \alpha_i y_i \mathbf{x}_i = 0 \;\Rightarrow\; \mathbf{w} = \sum_{i=1}^{\ell} \alpha_i y_i \mathbf{x}_i \quad (19)$$

$$\frac{\partial L(\mathbf{w}, b, \alpha)}{\partial b} = -\sum_{i=1}^{\ell} \alpha_i y_i = 0 \quad \Rightarrow \quad \sum_{i=1}^{\ell} \alpha_i y_i = 0 \quad (20)$$

By substituting these expressions back into L, we eliminate the primal variables and obtain the *Wolfe dual* of the optimization problem:

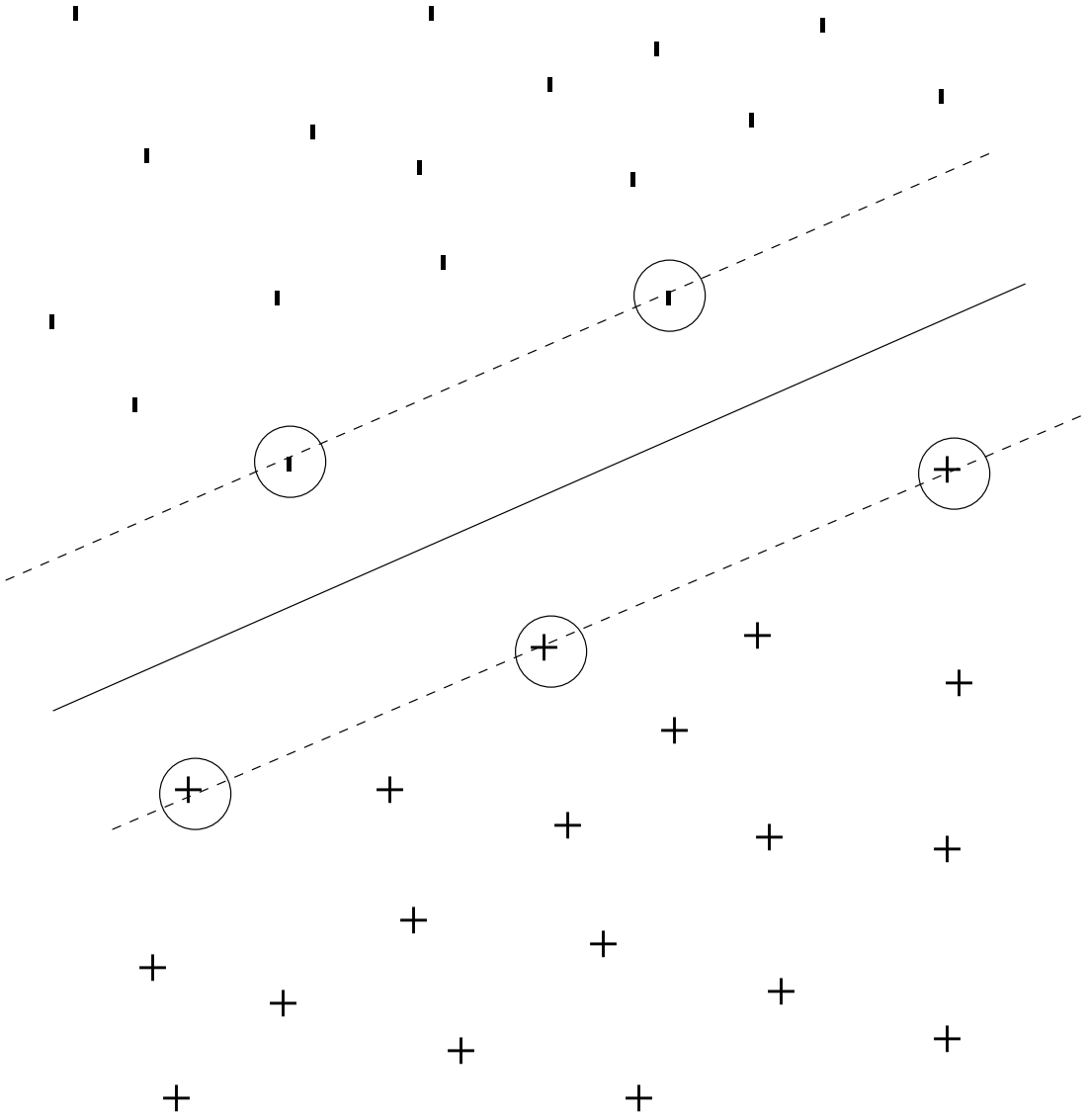
$$\begin{aligned} \text{max } & W(\alpha) = \sum_{i=1}^{\ell} \alpha_i - \frac{1}{2} \sum_{i=1}^{\ell} \sum_{j=1}^{\ell} \alpha_i \alpha_j y_i y_j (\mathbf{x}_i^T \mathbf{x}_j) \\ \text{s.t.} & \\ & \sum_{i=1}^{\ell} \alpha_i y_i = 0 \\ & \alpha_i \geq 0, \ \ i = 1, \ldots, \ell \end{aligned} \tag{21}$$

Let $(\mathbf{w}^*, b^*)$ be an optimal solution to the original optimization problem (17) and let α^* be an optimal solution to the above dual problem. By the KKT complementarity conditions, we have

$$\alpha_i^*[y_i(\mathbf{w}^{*T}\mathbf{x}_i + b^*) - 1] = 0, \ i = 1, \ldots, \ell \qquad (22)$$

The vectors $\mathbf{x}_i$ for which $\alpha_i^* > 0$ are called *support vectors*.

Note that the KKT complementarity conditions imply that the constraint in (17) is active for support vectors. That is, the support vectors lie on the margin of the optimal hyperplane.



α^* can be computed by solving the Wolfe dual. Moreover, from equation (22), it follows that b^* can be computed as

$$b^* = y_i - \mathbf{w}^{*T}\mathbf{x}_i. \tag{23}$$

Finally, the optimal hyperplane decision function is given by

$$f_{\mathbf{w}^*,b^*}(\mathbf{x}) = \text{sgn}(\mathbf{w}^{*T}\mathbf{x} + b^*)$$

$$= \text{sgn}\left(\sum_{i=1}^{\ell} y_i \alpha_i^* (\mathbf{x}^T \mathbf{x}_i) + b^*\right) \tag{24}$$

4.2 Linearly Non-Separable Case

Next, we deal with the case where a separating hyperplane does not exist.

As before, we assume that for any hyperplane $\mathbf{w}^T\mathbf{x}+b=0$, $\mathbf{w}$ and $\mathbf{x}$ are scaled so that the closest data point has distance $1/||\mathbf{w}||$.

Basic Idea: Relax the original separation constraints

$$y_i(\mathbf{w}^T\mathbf{x}_i + b) \geq 1, \quad i = 1, \ldots, \ell \tag{25}$$

but only when necessary, i.e. allow violations of the constraints but discourage it by introducing a penalty for each violation.

Introduce *slack variables*

$$\xi_i \geq 0, \quad i = 1, \ldots, \ell \tag{26}$$

that measure the amount of violation of the separation constraints.

The optimization problem becomes

$$\begin{aligned} &\min_{\mathbf{w},b,\xi} \ \frac{1}{2}||\mathbf{w}||^2 + C\sum_{i=1}^{\ell}\xi_i \\ &\text{s.t.} \\ &\quad y_i(\mathbf{w}^T\mathbf{x}_i + b) \geq 1 - \xi_i, \quad i = 1, \ldots, \ell \\ &\quad \xi_i \geq 0, \quad i = 1, \ldots, \ell \end{aligned} \tag{27}$$

where C is a parameter which has to be determined beforehand.

Remark: For any feasible solution $(\mathbf{w}, b, \xi)$ to the above optimization problem, $\sum_{i=1}^{\ell} \xi_i$ is an upper bound on the misclassification error.

Why? If the classification of $\mathbf{x}_i$ is in error, then we must have $\xi_i > 1$ because of the way $\mathbf{w}$ and b are scaled.

Note: We don't necessarily have an error when $\xi_i > 1$ for some feasible solution $(\mathbf{w}, b, \xi)$. But in an optimal solution, $\xi_i > 1$ implies that the data point $\mathbf{x}_i$ is misclassified.

Note that the above problem is again a quadratic programming problem and we could proceed in a manner similar to the linearly separable case.

The Lagrangian is given by $L(\mathbf{w}, b, \xi, \alpha, \gamma) =$

$$\frac{1}{2}||\mathbf{w}||^2 - \sum_{i=1}^{\ell} \alpha_i [y_i(\mathbf{w}^T \mathbf{x}_i + b) - 1 + \xi_i] - \sum_{i=1}^{\ell} \gamma_i \xi_i + C \sum_{i=1}^{\ell} \xi_i \quad (28)$$

where $\alpha = (\alpha_1, \ldots, \alpha_\ell)^T$ and $\gamma = (\gamma_1, \ldots, \gamma_\ell)^T$ are the vectors of nonnegative Lagrange multipliers corresponding to the constraints in (27).

As before, the optimization problem can be solved by finding a saddle point of L which minimizes L with respect to the primal variables $\mathbf{w} = (w_1, \ldots, w_d)^T$, $\xi = (\xi_1, \ldots, \xi_\ell)^T$, b and maximizes L with respect to the dual variables $\alpha = (\alpha_1, \ldots, \alpha_\ell)^T$ and $\gamma = (\gamma_1, \ldots, \gamma_\ell)^T$.

Differentiating L with respect to the primal variables and setting the results equal to zero, we get:

$$\frac{\partial L(\mathbf{w}, b, \xi, \alpha, \gamma)}{\partial \mathbf{w}} = \mathbf{w} - \sum_{i=1}^{\ell} \alpha_i y_i \mathbf{x}_i = 0$$

$$\Rightarrow \quad \mathbf{w} = \sum_{i=1}^{\ell} \alpha_i y_i \mathbf{x}_i \tag{29}$$

$$\frac{\partial L(\mathbf{w}, b, \xi, \alpha, \gamma)}{\partial b} = -\sum_{i=1}^{\ell} \alpha_i y_i = 0 \quad \Rightarrow \quad \sum_{i=1}^{\ell} \alpha_i y_i = 0 \tag{30}$$

$$\frac{\partial L(\mathbf{w}, b, \xi, \alpha, \gamma)}{\partial \xi_i} = C - \alpha_i - \gamma_i = 0$$

$$\Rightarrow \quad \alpha_i + \gamma_i = C, \;\; i = 1, \ldots, \ell \tag{31}$$

By substituting these expressions back into L, we again eliminate the primal variables and obtain the Wolfe dual:

$$\text{max} \;\; W(\alpha) = \sum_{i=1}^{\ell} \alpha_i - \frac{1}{2} \sum_{i=1}^{\ell} \sum_{j=1}^{\ell} \alpha_i \alpha_j y_i y_j (\mathbf{x}_i^T \mathbf{x}_j)$$

s.t. (32)

$$\sum_{i=1}^{\ell} \alpha_i y_i = 0$$

$$0 \leq \alpha_i \leq C, \;\; i = 1, \ldots, \ell$$

Let $(\mathbf{w}^*, b^*, \xi^*)$ be an optimal solution to the primal optimization problem (27) and let (α^*, γ^*) be an optimal solution to the above dual problem. By the KKT complementarity conditions, we have

$$\alpha_i^*[y_i(\mathbf{w}^{*T}\mathbf{x}_i + b^*) - 1 + \xi_i^*] = 0, \ \ i = 1, \ldots, \ell \qquad (33)$$

$$\gamma_i^* \xi_i^* = 0, \ \ i = 1, \ldots, \ell \qquad (34)$$

As before, the vectors $\mathbf{x}_i$ for which $\alpha_i^* > 0$ are called *support vectors*.

From the KKT conditions, note that $\alpha_i^* > 0$ implies that

$$y_i(\mathbf{w}^{*T}\mathbf{x}_i + b^*) = 1 - \xi_i^* \tag{35}$$

Two possibilities:

(i) $\xi_i^* = 0$ ($\mathbf{x}_i$ is a support vector that lies on the margin of the hyperplane); or

(ii) $\xi_i^* > 0$ ($\mathbf{x}_i$ is a misclassified support vector).

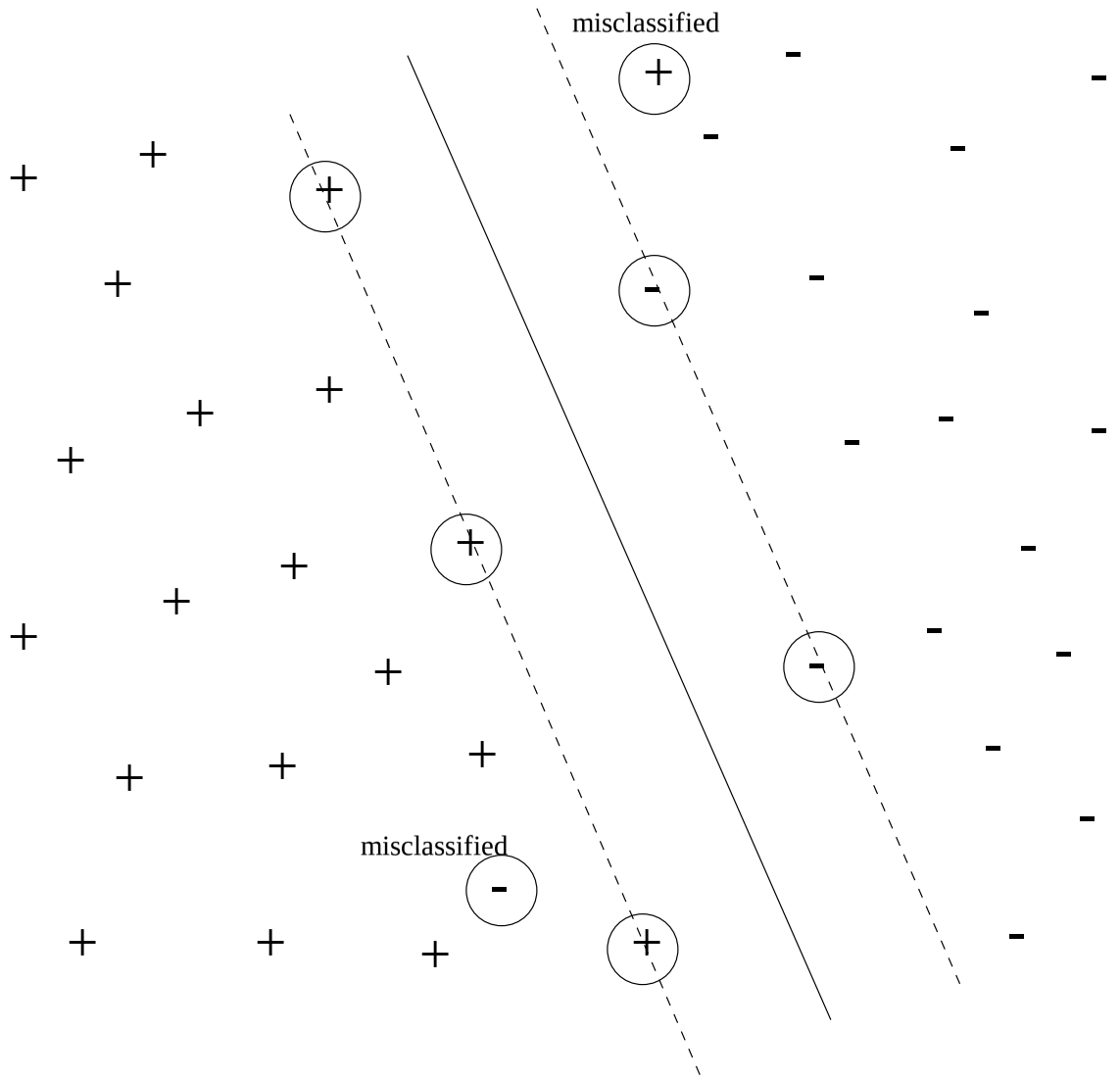
Note that

$$0 < \alpha_i^* < C \Rightarrow C - \gamma_i^* = \alpha_i^* < C \Rightarrow \gamma_i^* > 0 \Rightarrow \xi_i^* = 0 \quad (36)$$

Hence, if $0 < \alpha_i^* < C$, then x_i is a support vector that lies on the margin of the optimal hyperplane. However, if $\alpha_i^* = C$, then it does not follow that x_i is a misclassified support vector.

Remark: Any data point that is misclassified by an optimal hyperplane is a support vector.

Why? If $\mathbf{x}_i$ is misclassfied, then we must have $\xi_i^* > 0$. Then by the KKT conditions, $\gamma_i^* = 0$, and so, $\alpha_i^* = C > 0$.


misclassified
misclassified

As before, α^* can be computed by solving the Wolfe dual. Moreover, from equation (33), it follows that b^* can be computed by considering a support vector $\mathbf{x}_i$ such that $0 < \alpha_i^* < C$ which gives

$$b^* = y_i - \mathbf{w}^{*T}\mathbf{x}_i. \tag{37}$$

Again, the optimal hyperplane decision function is given by

$$f_{\mathbf{w}^*,b^*}(x) = \text{sgn}(\mathbf{w}^{*T}\mathbf{x} + b^*)$$

$$= \text{sgn}\left(\sum_{i=1}^{\ell} y_i \alpha_i^* (\mathbf{x}^T \mathbf{x}_i) + b^*\right) \tag{38}$$

5. Nonlinear Support Vector Machines

In most cases, linear decision surfaces are not appropriate for many classification tasks. So how do we extend the previous method to deal with more complex (i.e. nonlinear) decision surfaces?

Basic Idea: Nonlinearly transform the set of input vectors into a high (possibly infinite) dimensional feature space and perform a linear separation there.

Introduce a map $\Phi : R^d \to F$, where F is a high dimensional feature space that is equipped with a dot product. Hence, the training problem becomes

$$\max \; W(\alpha) = \sum_{i=1}^{\ell} \alpha_i - \frac{1}{2} \sum_{i=1}^{\ell} \sum_{j=1}^{\ell} \alpha_i \alpha_j y_i y_j < \Phi(\mathbf{x}_i), \Phi(\mathbf{x}_j) >$$

s.t.

$$\sum_{i=1}^{\ell} \alpha_i y_i = 0 \tag{39}$$

$$0 \leq \alpha_i \leq C, \;\; i = 1, \ldots, \ell$$

Computing dot products in F can be expensive. However, these calculations can be reduced significantly if there is a function K such that

$$< \Phi(\mathbf{x}), \Phi(\mathbf{z}) >= K(\mathbf{x}, \mathbf{z}) \tag{40}$$

Such a function is called a *kernel*.

Assuming that a kernel function K exists, the training problem is now

$$\max \; W(\alpha) = \sum_{i=1}^{\ell} \alpha_i - \frac{1}{2} \sum_{i=1}^{\ell} \sum_{j=1}^{\ell} \alpha_i \alpha_j y_i y_j K(\mathbf{x}_i, \mathbf{x}_j)$$

s.t. (41)

$$\sum_{i=1}^{\ell} \alpha_i y_i = 0$$

$$0 \leq \alpha_i \leq C, \; i = 1, \ldots, \ell$$

leading to the decision function

$$f(x) = \text{sgn}(\mathbf{w}^{*T}\mathbf{x} + b^*) = \text{sgn}\left(\sum_{i=1}^{\ell} y_i \alpha_i^* K(\mathbf{x}, \mathbf{x}_i) + b^*\right) \quad (42)$$

where α^* is an optimal solution to (41) and b^* is obtained as before.

Now the obvious questions is: How do we find F, a map $\Phi : R^d \rightarrow F$, and when do we have a kernel K with the above property?

It is actually easier to begin with a function $K : R^d \times R^d \rightarrow R$ and determine when a mapping $\Phi : R^d \rightarrow F$ exists such that

$$< \Phi(\mathbf{x}), \Phi(\mathbf{z}) >= K(\mathbf{x}, \mathbf{z}) \tag{43}$$

The answer is given by Mercer's condition.

(Mercer's Condition): There exists a mapping Φ and an expansion

$$K(\mathbf{x}, \mathbf{z}) = \sum_i \Phi(x)_i \Phi(z)_i \tag{44}$$

if and only if, for any $g(x)$ such that

$$\int g(\mathbf{x})^2 d\mathbf{x} \text{ is finite} \tag{45}$$

then

$$\int K(\mathbf{x}, \mathbf{z}) g(\mathbf{x}) g(\mathbf{z}) d\mathbf{x} d\mathbf{z} \geq 0. \tag{46}$$

Note that as long as we have a kernel function with the above property, we don't really need to know F and Φ. All the training and classification can be done via the kernel.

Examples:

Polynomial classifiers of degree d:

$$K(\mathbf{x}, \mathbf{z}) = (< \mathbf{x}, \mathbf{z} > +1)^d \tag{47}$$

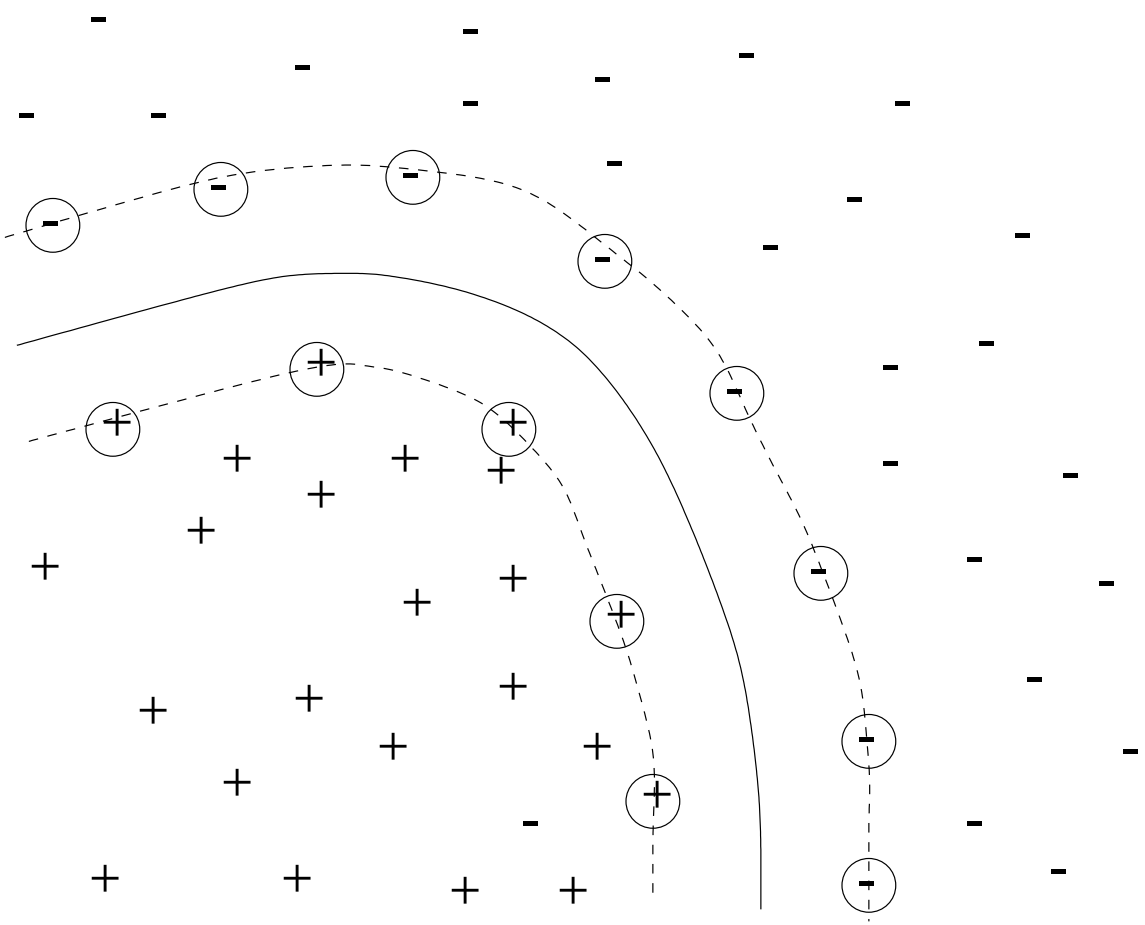
Radial basis function classifiers:

$$K(\mathbf{x}, \mathbf{z}) = \exp(-||\mathbf{x} - \mathbf{z}||^2/2\sigma^2) \tag{48}$$

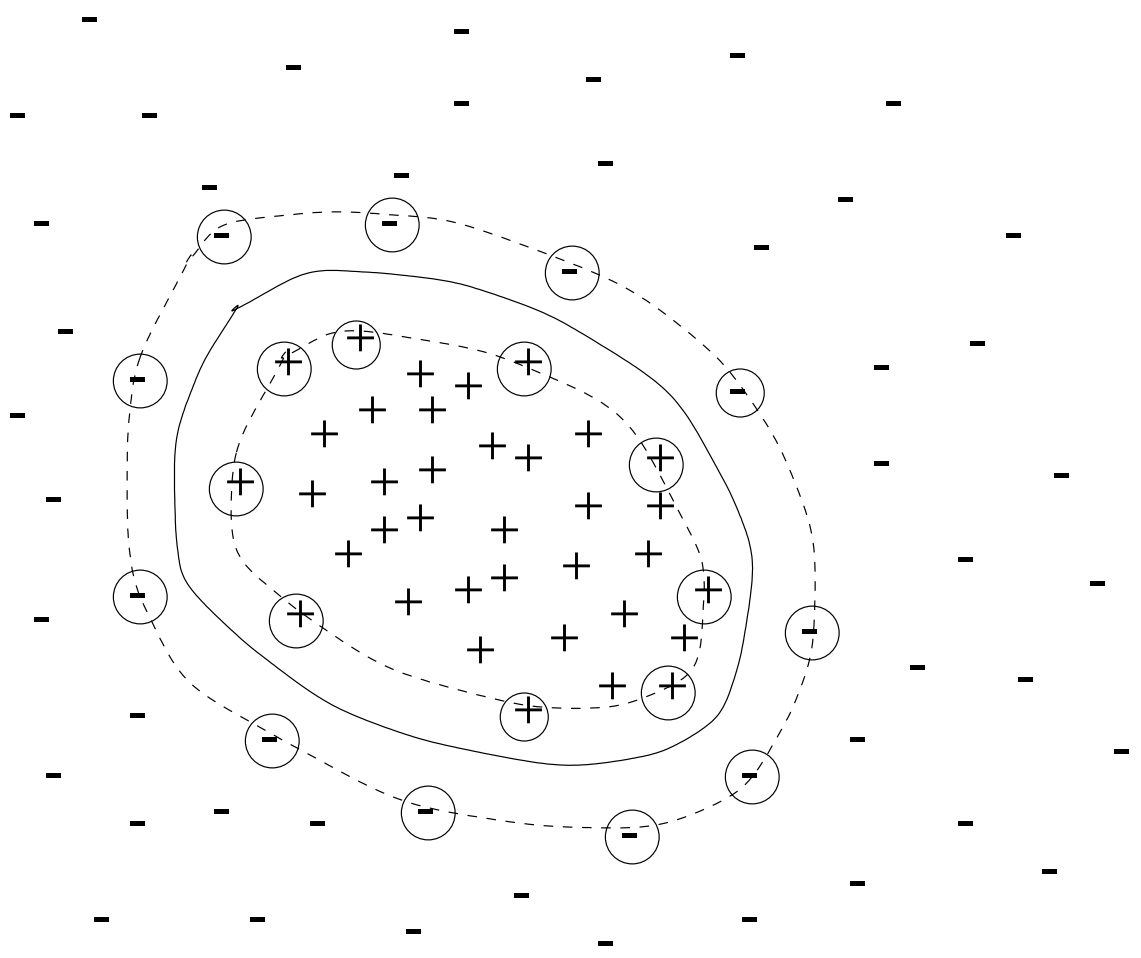
Neural networks:

$$K(\mathbf{x}, \mathbf{z}) = \tanh(\kappa < \mathbf{x}, \mathbf{z} > +\theta) \tag{49}$$

polynomial classifier



radial basis function (RBF) classifier



6. References and Further Reading:

1. Burges, C.J.C. (1998), A Tutorial on Support Vector Machines for Pattern Recognition, *Data Mining and Knowledge Discovery* 2(2): 955-974.

2. Campbell, C. (2000), An Introduction to Kernel Methods, in: Howlett, R.J. and Jain, L.C. (eds), *Radial Basis Function Networks: Design and Applications*, Berlin, Springer Verlag.

3. Cristianini, N. and Shawe-Taylor, J. (2000), *Introduction to Support Vector Machines*, Cambridge University Press.

4. Osuna, E., Freund, R. and Girosi, F. (1997), Support Vector Machines: Training and Applications

5. Scholköpf, B. (1997), Support Vector Learning, *PhD Thesis*, Technische Universita Berlin, Berlin, Germany.

6. Vapnik, V. (1998), *Statistical Learning Theory*, Wiley-Interscience.