6. Support vector machines

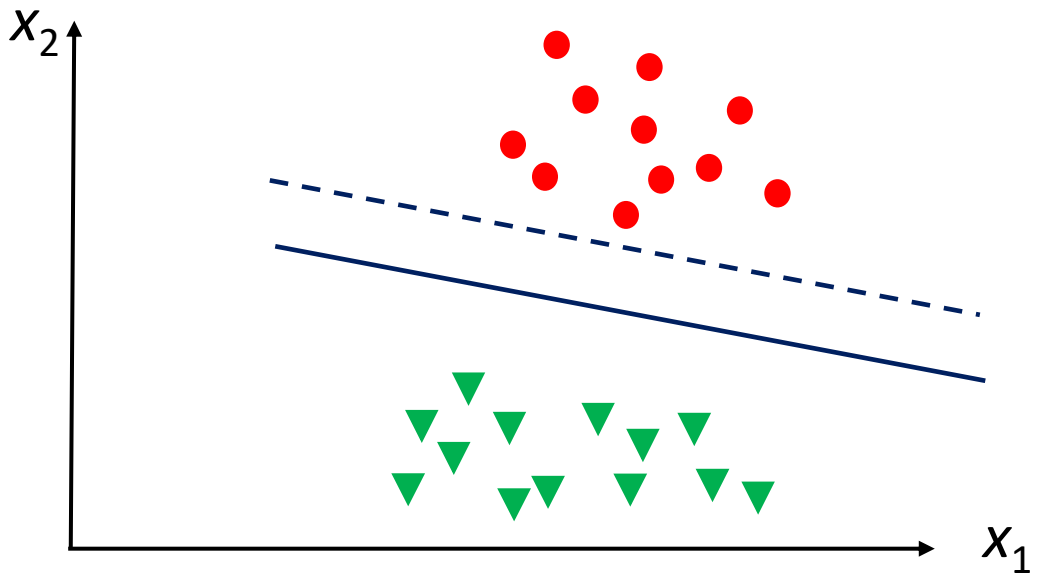


Fig. 6.1 A linearly separable two class problem with two possible hyperplanes.

Support vector machines implement a simple idea to map data case vectors in the variable space where a ”best” separating hyperplane – the maximal margin hyperplane – is constructed. Fig. 6.1 and Fig. 6.2 represent this.

6.1 Linearly separable data (classes)

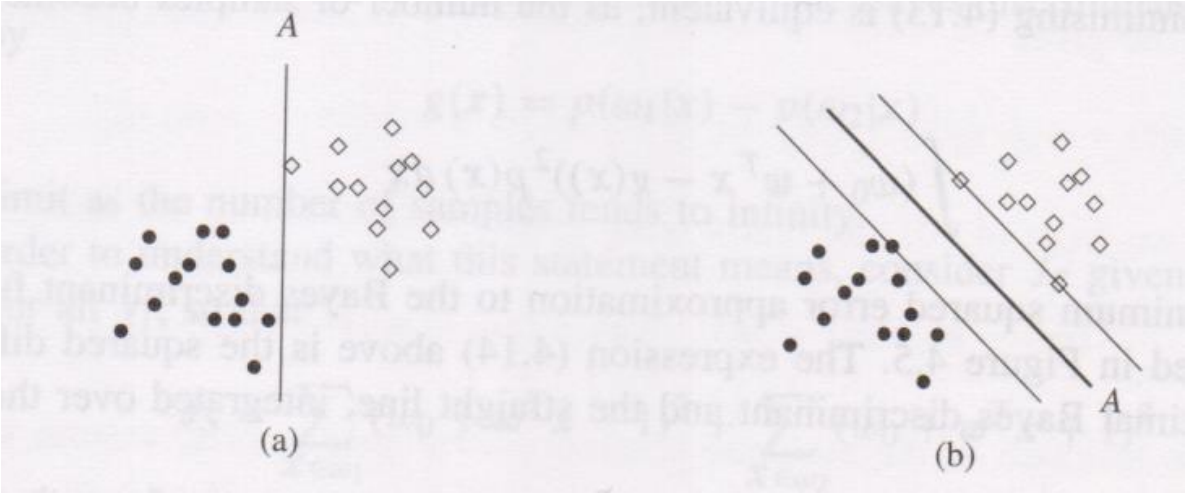


Fig. 6.2 Two linearly separable data sets with a separating hyperplane. (a) The hyperplane parallel with the vertical axis is a poor choice being close to some data points. (b) The (thick) separating hyperplane leaves the closest points at maximum distance. The thin lines identify the margin.

Let $\mathbf{x}_i$, i=1,2,…, N be the variable vectors of the training set L. These belong to two classes, c_1 or c_2, which are assumed to be linearly separable. The goal is to find a discriminant function

$$g(\mathbf{x}) = \mathbf{w}^{\mathrm{T}}\mathbf{x} + w_0 \qquad (6.1)$$

so that the hyperplane

$$g(\mathbf{x}) = \mathbf{w}^{\mathrm{T}}\mathbf{x} + w_0 = 0$$

classifies correctly all training cases. The hyperplane is not necessarily unique as Figs. 6.1. and 6.2 denote.

From formula (6.1) we obtain decision rule:

$$\mathbf{w}^{\mathrm{T}}\mathbf{x}_i + w_0 > 0, \forall \mathbf{x} \in c_1 \text{ with corresponding numeric value } \mathrm{y_i} = +1$$
$$\mathbf{w}^{\mathrm{T}}\mathbf{x}_i + w_0 < 0, \forall \mathbf{x} \in c_2 \text{ with corresponding numeric value } \mathrm{y_i} = -1$$

All training points are correctly classified if

$$y_i\left(\mathbf{w}^{\mathrm{T}}\mathbf{x}_i + w_0\right) > 0$$

for all *i.*

Which hyperplane in Fig. 6.1 would be a better choice? No doubt the full-line is better, because it leaves more ”room” on either side, so that data in both classes can move a bit more freely, with less risk of causing an error. Such a hyperplane can be trusted more, when it faced with unknown test data. This touches the important issue of classifier design: *the generalization performance of the classifier.* This refers to the capability of the classifier, designed using the training set, to operate satisfactorily with data outside this set.

The maximal margin classifier determines the hyperplane for which the *margin* – the distance to two parallel hyperplanes on each side of the hyperplane that separates the data – is the largest (Fig. 6.2.(b)).

Every hyperplane is characterized by its direction determined by vector **w** and its exact position determined by scalar w_0 in variable space. Since no preference is given to either of the classes, then it is reasonable for each direction to select that hyperplane which has the same distance from the respective nearest points in c_1 and c_2 according to Fig. 6.3. The hyperplanes in directions 1 and 2 are chosen from the infinite set. The margin for direction 1 is $2z_1$ and that for direction 2 is $2z_2$. *The objective is to search for the direction that gives the maximum possible margin.*

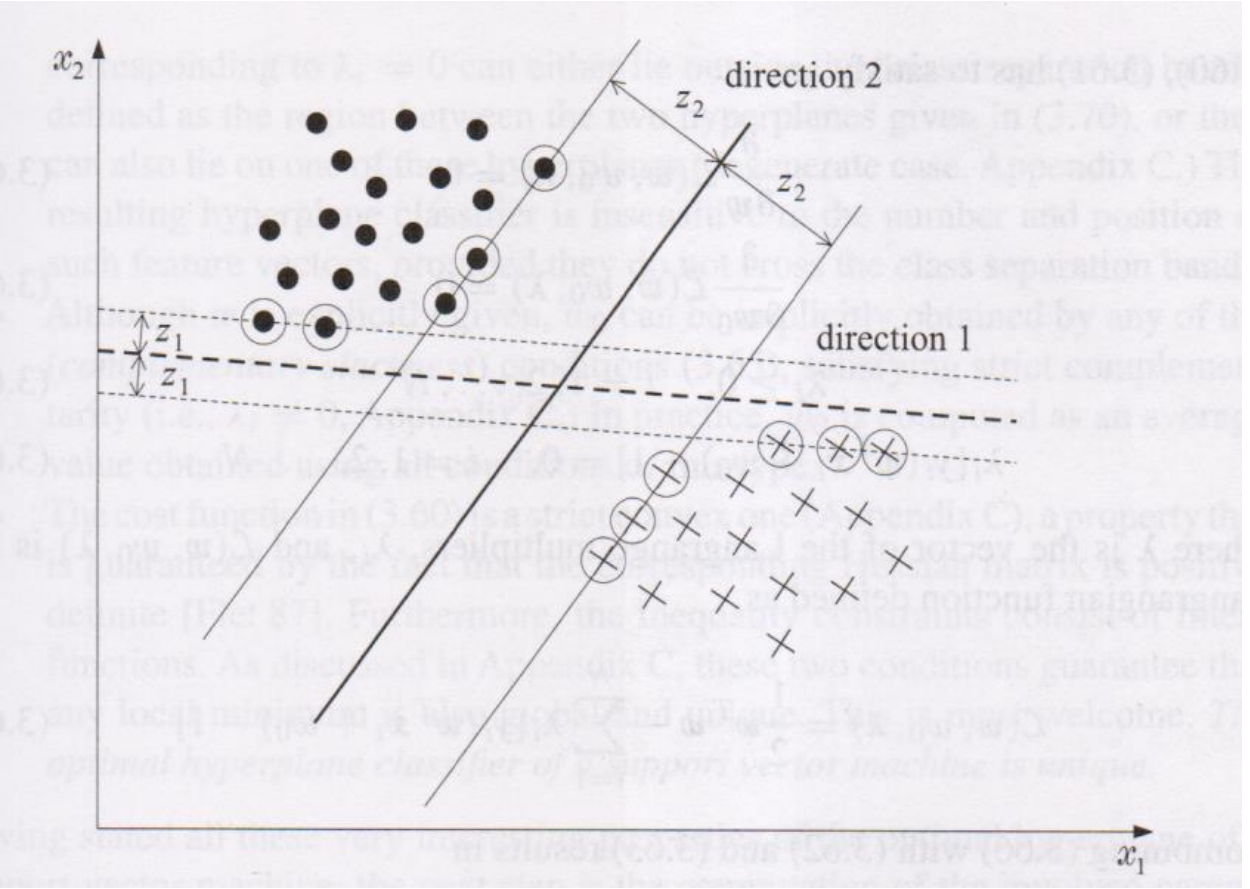


Fig. 6.3 The margin for direction 2 is larger, so better, than that for direction 1.

Each hyperplane is determined within a scaling factor. We will free ourselves from it, by appropriate scaling of all candidate hyperplanes. We remember that the distance of a point from a hyperplane is given by the following ratio:

$$z = \frac{|g(\mathbf{x})|}{\|\mathbf{w}\|}$$

Now we can scale $\mathbf{w}$, w_0 so that the value of $g(\mathbf{x})$, at the nearest points in c_1, c_2 (circled in Fig. 6.3), is equal to 1 for c_1 and, thus, equal to -1 for c_2. This is equivalent with

1. having a margin of $$\frac{1}{\|\mathbf{w}\|} + \frac{1}{\|\mathbf{w}\|} = \frac{2}{\|\mathbf{w}\|}$$

2. requiring that $$\mathbf{w}^{\mathrm{T}}\mathbf{x} + w_0 \geq 1, \forall \mathbf{x} \in c_1$$
$$\mathbf{w}^{\mathrm{T}}\mathbf{x} + w_0 \leq -1, \forall \mathbf{x} \in c_2$$

Now a brief formulation is given just to show how a nonlinear (quadratic) optimization task is here solved. We denote the class indicator or label by y_i= +1 for c_1 and -1 for c_2. The task is now: compute the parameters **w**, w_0 of the hyperplane so that to:

$$\text{minimize} \qquad J(\mathbf{w}) = \frac{1}{2}\|\mathbf{w}\|^2 \tag{6.2}$$

$$\text{subject to constraints} \qquad y_i\left(\mathbf{w}^{\mathrm{T}}\mathbf{x}_i + w_0\right) \geq 1, i = 1,2,\ldots,N$$

Minimizing the norm makes the margin maximum. This optimization is is subject to a set of linear constraints. The Karush-Kuhn-Tucker conditions that the minimizer of (6.2) has to satisfy are

$$\frac{\partial}{\partial \mathbf{w}} \mathrm{L}(\mathbf{w}, w_0, \boldsymbol{\lambda}) = \mathbf{0} \tag{6.3}$$

$$\frac{\partial}{\partial w_0} \mathrm{L}(\mathbf{w}, w_0, \boldsymbol{\lambda}) = 0 \tag{6.4}$$

$$\lambda_i \geq 0, i = 1,2,\ldots,N$$
$$\lambda_i \left(y_i \left(\mathbf{w}^{\mathrm{T}} \mathbf{x}_i + w_0 \right) - 1 \right) = 0, i = 1,2,\ldots,N \tag{6.5}$$

where $\boldsymbol{\lambda}$ is the vector of the Lagrange multipliers, λ_i, and $\mathrm{L}(\mathbf{w}, w_0, \boldsymbol{\lambda})$ is the Lagrangian function defined as

$$\mathrm{L}(\mathbf{w}, w_0, \boldsymbol{\lambda}) = \frac{1}{2}\mathbf{w}^{\mathrm{T}}\mathbf{w} - \sum_{i=1}^{N} \lambda_i \left(y_i \left(\mathbf{w}^{\mathrm{T}}\mathbf{x}_i + w_0 \right) - 1 \right) \qquad (6.6)$$

Combining (6.6) with (6.3) and (6.4) results in:

$$\mathbf{w} = \sum_{i=1}^{N} \lambda_i y_i \mathbf{x}_i$$

$$\sum_{i=1}^{N} \lambda_i y_i = 0$$

The vector parameter **w** of the optimal solution is a linear combination of $N_s<N$ variable vector (some of λ_i may be equal to 0), which gives the following form.

$$\mathbf{w} = \sum_{i=1}^{N_s} \lambda_i y_i \mathbf{x}_i$$

These are known as *support vectors* and the optimum hyperplane classifier as a *support vector machine* (SVM). As the set of constraints (6.5) suggest for $\lambda_i \neq 0$, the *support vectors lie on either of the two hyperplanes*, i.e., they are the training vectors that are closest to the linear classifier. See Fig. 6.4.

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

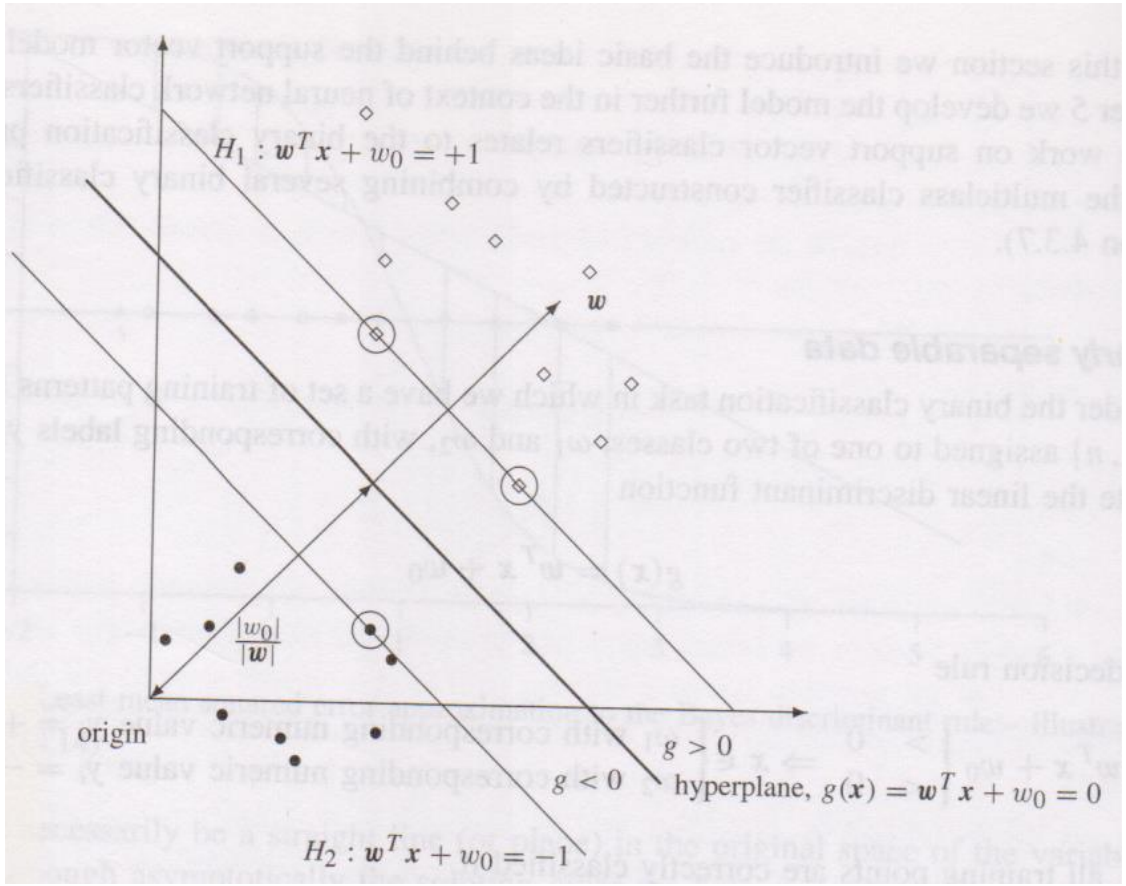


Fig. 6.4 H_1 and H_2 are the canonical hyperplanes. The margin is the perpendicular distance between the separating hyperplane $g(\mathbf{x})=0$ and a hyperplane through the closest points marked with rings around them. These are *support vectors*.

6.2 Nonseparable classes

In the situation where the classes are not separable, the previous setup is not valid any more. Fig. 6.5 illustrates this. Any attempt to draw a hyperplane will never end up with a class separation band with no data points inside it, as was the case in the linearly separable case.

The training vectors now belong to one of the following three categories:

- Vectors that fall outside the band and are correctly classified. These vectors comply with the constraints of (6.2).

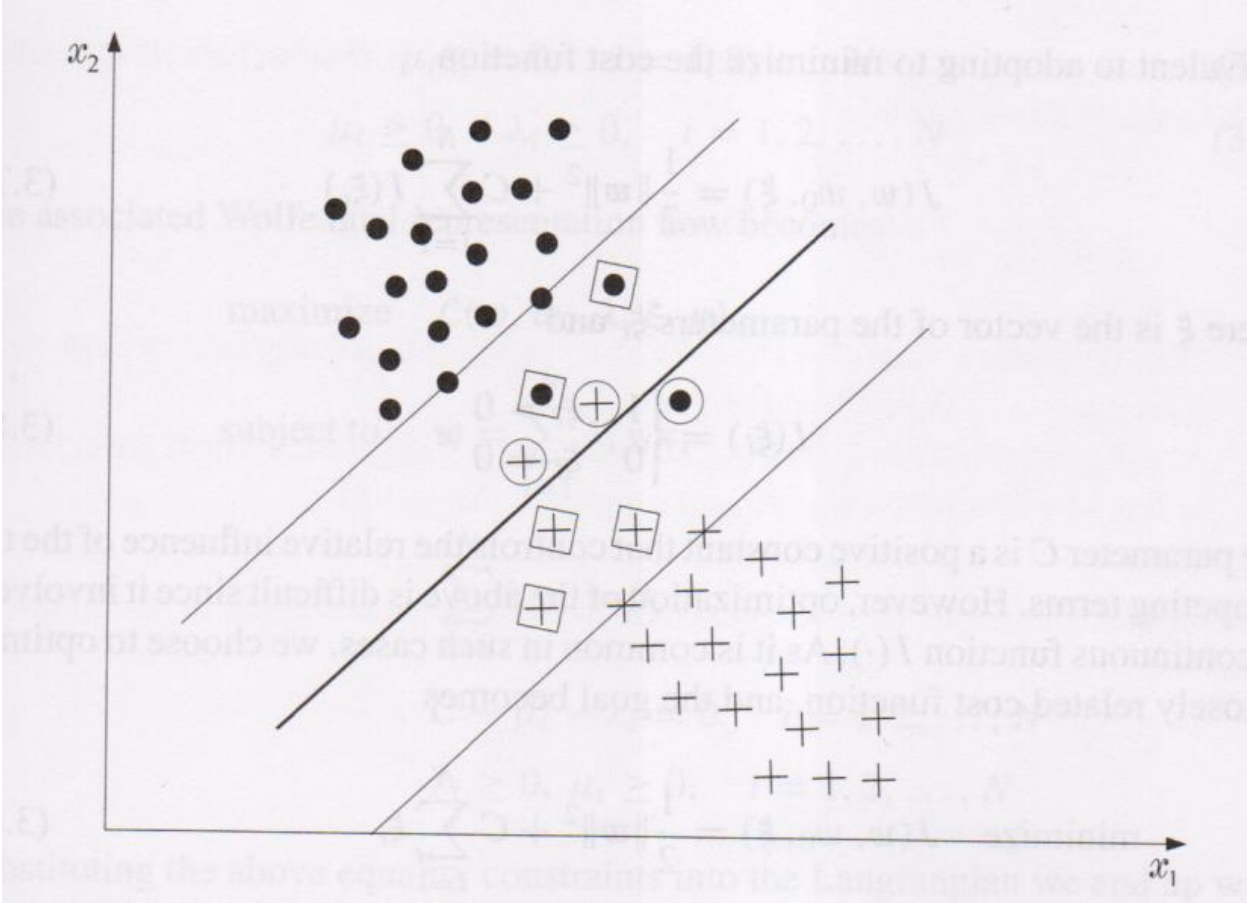


Fig. 6.5 In the nonseparable class situation, points fall inside the class separation band.

- Vectors falling inside the band and which are correctly classified. These are the points placed in squares in Fig. 6.5 and they satisfy the inequality:

$$0 \leq y_i\left(\mathbf{w}^{\mathrm{T}}\mathbf{x}_i + w_0\right) < 1$$

- Vectors that are misclassified. They are enclosed by circles and obey the inequality:

$$y_i\left(\mathbf{w}^{\mathrm{T}}\mathbf{x}_i + w_0\right) < 0$$

All three cases can be treated under a single type of constraints by introducing a new set of variables, namely.

$$y_i\left(\mathbf{w}^{\mathrm{T}}\mathbf{x}_i + w_0\right) \geq 1 - \xi_i$$

This results, instead of (6.2), in minimizing the cost function

$$J(\mathbf{w}, w_0, \boldsymbol{\xi}) = \frac{1}{2}\|\mathbf{w}\|^2 + Q\sum_{i=1}^{N} I(\xi_i)$$

where $\boldsymbol{\xi}$ is the vector of parameters ξ_i and Q a positive constant controling the relative influence of two competing terms. See Fig. 6.6.

$$I(\xi_i) = \begin{cases} 1 & \xi_i > 0 \\ 0 & \xi_i = 0 \end{cases}$$

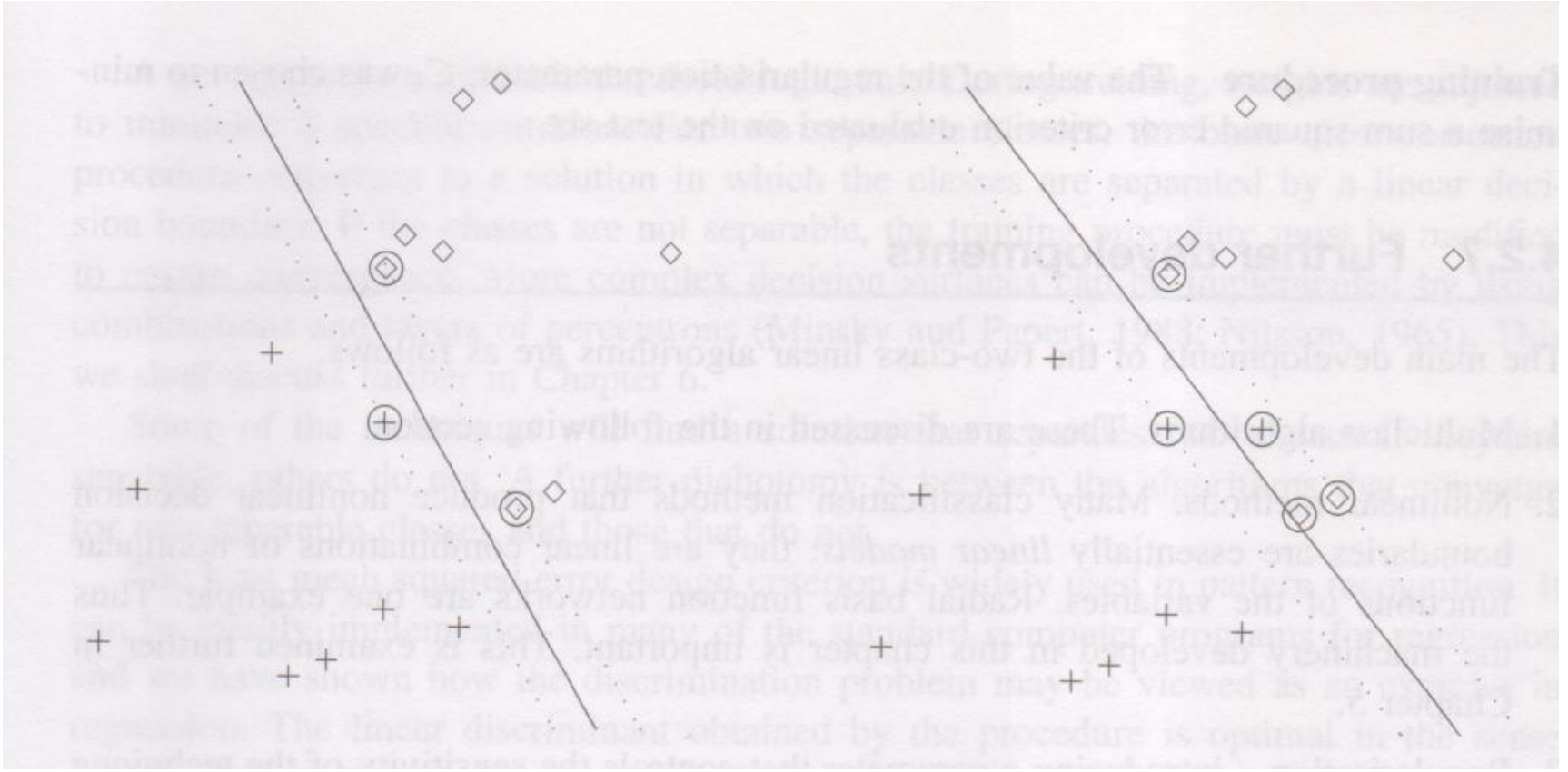


Fig. 6.6 Linearly separable data (left) and nonseparable (right, Q=20).

Note that we considered purely **linear** support vector machines. There are other kernel functions that may produce better results depending on data, such as **quadratic**, or **polynomials** of degree 3 or more, **Gaussian** and **radial basis function kernel** among others.

6.3 Multiclass classification

Thus far, classfication considered for support vector machines was of two classes only. There are several ways of extending two class procedures to the multiclass case.

We only mention two simplest alternatives. One is one-versus-all (OVA or one-versus-rest) in which C (the number of classes) binary classifiers are constructed. The other is one-versus-one (OVO) in which $C(C\text{-}1)/2$ binary classifiers are built for pairs of classes. The classification is based on majority voting. Their use was presented on p. 72-77.

Example 1: Demographic vs. crime variables

Returning again to the example [12] from p. 122, p. 143 and p. 177, we obtained results better than those given by naive Bayes, nearest neighbor searching or discriminant analysis. See Table 6.1 subject to classification accuracies (%) of support vector machines after scaling or without it. There are also results of other approaches than OVA and OVO.

[12] X. Li, H. Joutsijoki, J. Laurikkala and M. Juhola: Crime vs. demographic factors: application of data mining methods, Webology, Article 132, 12(1), 1-19, 2015.

Table 6.1 Method SVM	Not scaled	Scaled into [0,1]	Standardized
ECOC-OVA-SVM	Accuracy %	Accuracy %	Accuracy %
Linear	69.6	87.5	**94.6**
Polynomial degree 2	26.8	87.5	87.5
Polynomial degree 3	30.4	85.7	76.8
ECOC-OVO-SVM			
Linear	71.4	83.9	85.7
Polynomial degree 2	28.6	80.4	78.6
Polynomial degree 3	58.9	80.4	80.4
ECOC-ORD-SVM			
Linear	76.8	80.4	78.6
Polynomial degree 2	26.8	89.3	89.3
Polynomial degree 3	25.0	91.1	76.8
ECOC-BIN-SVM			
Linear	76.8	89.3	92.9
Polynomial degree 2	25.0	85.7	89.3
Polynomial degree 3	25.0	85.7	76.8

Example 2: Vertigo data

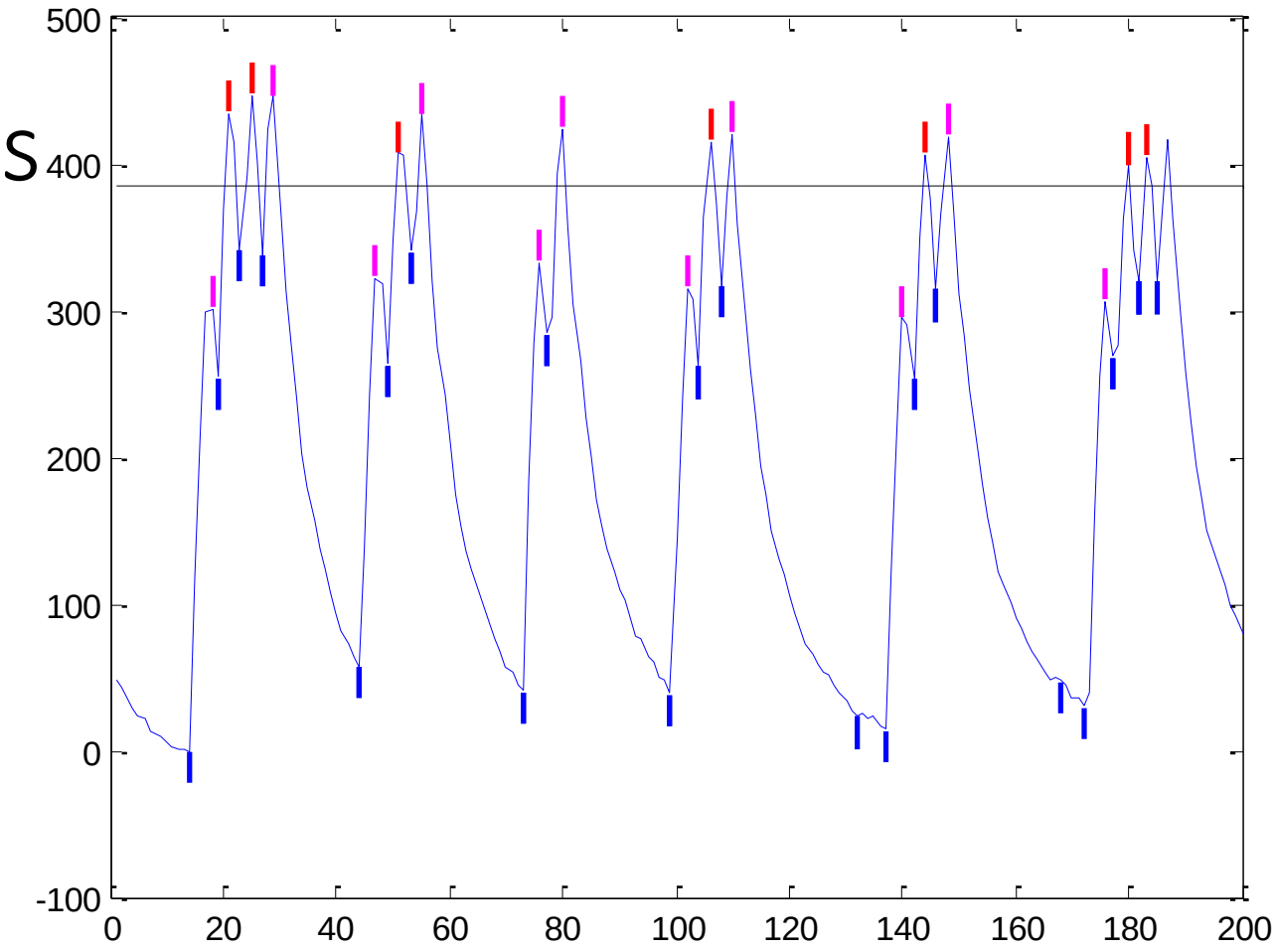
For Vertigo data[13] from p. 124-126, p. 146 and p. 178 we achieved the following results with support vector machines. For both quadratic function as kernel and for the radial basis function we obtained accuracy of 87.2%. This was slightly worse than that in Section 5, p. 178, but quite similar to those on p. 146.

(In Matlab support vector machine models are built with ’fitcsvm’ for binary classification. A model is then tested with ’predict’. For multiclassification there is ’fitcecoc’.)

[13] M. Juhola: Data Classification, Encyclopedia of Computer Science and Engineering, ed. B. Wah, John Wiley & Sons, 2008 (print version, 2009, Hoboken, NJ), 759-767.

Example 3: Cardiomyocytes

Fig. 6.7 Stem cell-derived cardio-myocytes (heart cells) in a laboratory culture basin were exposed to two different wavelengths of light and emissions recorded. For calcium transient analysis, regions of interest were selected from a video stream of spontaneously beating cells. A signal (mean removed) of around 15 s with all peaks recognized abnormal represents an abnormal calcium cycling waveform[14].



[14]M. Juhola et al., and K. Aalto-Setälä, Signal analysis and classification methods for the calcium transient data of stem cell-derived cardiomyocytes, Computers in Biology and Medicine, 61, 1-7, 2015.

In the following tables we see classification results (average sensitivity, specificty and accuracy) of leave-one-out test runs for 280 calcium transient signals measured from cardiomyocytes.

After having recognized peaks either to the normal or abnormal, entire signals were classified to be either normal or abnormal. If even one peak of a signal was found to be abnormal, it means that the whole signal is seen abnormal. Therefore, there are two alternatives: if all peaks were normal, the signal should be classified as normal. If one or more peaks were abnormal, the signal should be abnormal.

Two approaches were applied. (1) The signal detection program recognized peaks of each signal and determined them to be normal or abnormal. Then, this way, the final decision on the label of the signal was based on peak data. Thereafter, all signals were classified with different classifiers, and it was checked whether classifications produced by the classifiers were correct, i.e. the same as determined by the detection program.

(2) Instead of the class labels determined by the program to the signals, a human biotechnology expert defined the correct labels on the basis of which ckecks were made whether these were same as those given by classifiers.

Naturally, we could wait for that classification results obtained in approach (1) would be higher since the program produced (systematically) peak data with class labels for classification, but in (2) a human expert was a different (and not entirely systematic) ”actor” to determine just class labels of the signals. The peak data (variable values) were computed by the program in both situations.

Table 6.2 Signals classified by peaks and compared to the *computationally determined classes*: percentages of correct classification produced by *k*-nearest neighbor search, *k*=1, 3, …,21, and linear (LDA) and quadratic (QDA) discriminant analysis. Number *n* is for the normal and abnormal signals.

Quantity	*n*	1	3	5	7	9	11	13	15	17	19	21	LDA	QDA
Sensitivity	125	87.7	94.2	97.4	96.8	98.1	98.7	98.7	98.1	98.1	98.1	98.1	60.4	70.8
Specificity	155	88.9	84.9	72.2	69.8	66.7	64.3	63.5	61.9	61.1	58.7	55.6	92.9	96.0
Accuracy		88.2	**90.0**	86.1	84.6	83.9	83.2	82.9	81.8	81.4	80.4	78.9	75.0	82.1

Table 6.3 Signals classified by peaks and compared to the classes determined by the *human expert*: percentages of correct classification produced by *k*-nearest-neighbor search, *k*=1, 3, …,21, and linear (LDA) and quadratic (QDA) discriminant analysis. Number *n* is for the normal and abnormal signals.

Quantity	*n*	1	3	5	7	9	11	13	15	17	19	21	LDA	QDA
Sensitivity	125	84.0	90.4	91.2	92.8	92.8	92.8	92.8	93.6	93.6	93.6	93.6	60.8	72.8
Specificity	155	71.6	67.1	54.2	54.2	50.3	47.7	47.1	47.1	46.5	44.5	41.9	83.2	85.2
Accuracy		77.1	77.5	70.7	71.4	69.3	67.9	67.5	67.9	67.5	66.4	65.0	73.2	**79.6**

Table 6.4 Signals classified by peaks and compared to the classes determined by the *human expert*: percentages of correct classification produced by discriminant analysis using Mahalanobis distance (DAM), normal naïve Bayes (NBN) and Bayes with kernel (NBK), classification tree, and self-organizing maps of sizes of 10×10 (SOM1), 20×20 (SOM2) and 30×30 (SOM3) nodes. Number *n* is for the normal and abnormal signals.

Quantity	*n*	DAM	NBN	NBK	Tree	SOM1	SOM2	SOM3
Sensitivity	125	46.4	68.8	88.0	72.0	91.0	79.8	72.2
Specificity	155	99.3	64.5	43.9	74.8	29.1	55.4	67.7
Accuracy		71.8	66.4	63.6	**73.6**	56.7	66.3	69.7

Table 6.5 Signals classified by peaks and compared to the classes determined by the *human expert*: percentages of correct classification produced by support vector machines with linear kernel (LIN), quadratic kernel (QUAD), 3rd degree kernel (3DEG), 4th degree kernel (4DEG), 5th degree kernel (5DEG), 6th degree kernel (6DEG), and radial basis function kernel (RBF). Number *n* is for the normal and abnormal signals.

Quantity	*n*	LIN	QUAD	3DEG	4DEG	5DEG	6DEG	RBF
Sensitivity	125	51.2	54.4	54.4	50.4	48.8	40.0	56.0
Specificity	155	82.6	89.7	94.8	91.6	92.3	94.8	90.3
Accuracy		68.6	73.9	**76.8**	73.2	72.9	70.4	75.0