SUPPORT VECTOR MACHINE CLASSIFICATION

M.P. Wand[1]

18th July, 2006

Support vector machines emerged in the mid-1990s as a flexible and powerful means of classification. Classification is a very old problem in Statistics but, in our increasingly data-rich age, remains as important as ever. Some examples of classification are:

- classify a patient as high or low risk of prostate cancer based on personal attributes and some medical measurements;
- classify an e-mail message as 'spam' or normal based on features in the text such as the frequency of capital letters;
- classify a person as an employee or intruder at a workplace site based on a retina scan.

One area of classification that has led to an enormous amount of research is computer-aided mail sorting. Figure 1 shows 3 sets of handwritten digits. Suppose that some of these appeared on the post code of an addressed envelope. How might we get computers to help us 'read' the post code?

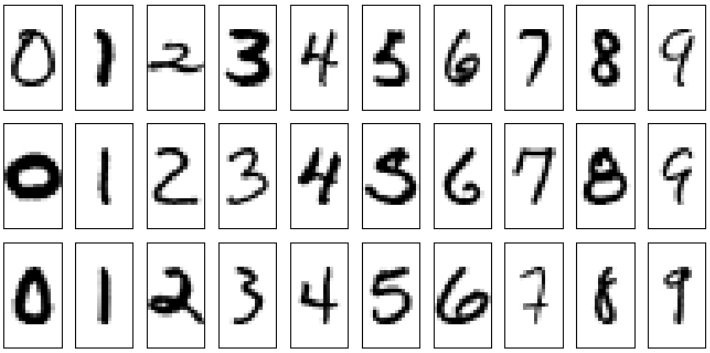


Figure 1: *Three sets of handwritten digits.*

Support vector machines are an effective means of dealing with such classification problems. They are also relatively simple. Indeed, the two ingredients of support vector machines: *maximum margin lines* (and their extension to higher-dimensions) and *kernelisation*, can be explained using mainly high school-level mathematics.

Maximum margin lines. Classification involves the use of *training data* to construct rules for classification of future observations based on their *features*. We can think of training data as points in high-dimensional space, with the points coloured according to the known class. Figure 2 shows a small two-dimensional training data set. The two classes correspond to the black and white colouring. We will now describe a method, known as *maximum margin lines*, for classifying future points as either 'black' or 'white'.

Suppose that we restrict the classifier to the family of straight lines. Then Figure 3 shows three lines that provide perfect separation. Clearly there are an infinite number of such lines, but which one is 'best'? One way to define 'best' is via the concept of *maximising the margin*. For any separating line we can form a *margin region* by considering rectangles that are bisected by the line and do not contain any of the training data points. We then find the line that produces the widest rectangle.

[1]M.P. Wand is a Professor of Statistics, School of Mathematics and Statistics, The University of New South Wales.

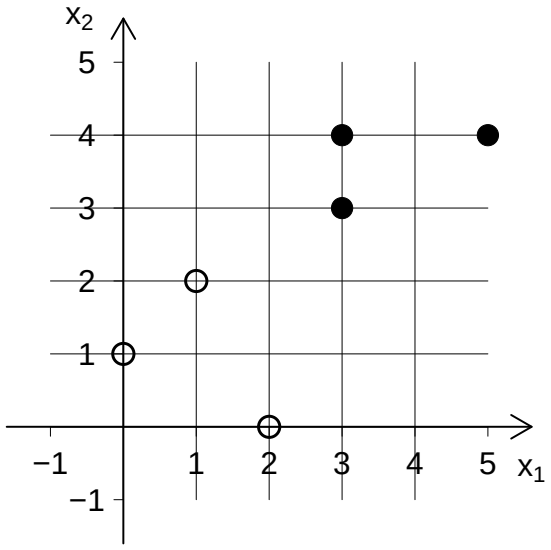


Figure 2: *An example two-dimensional training data set.*

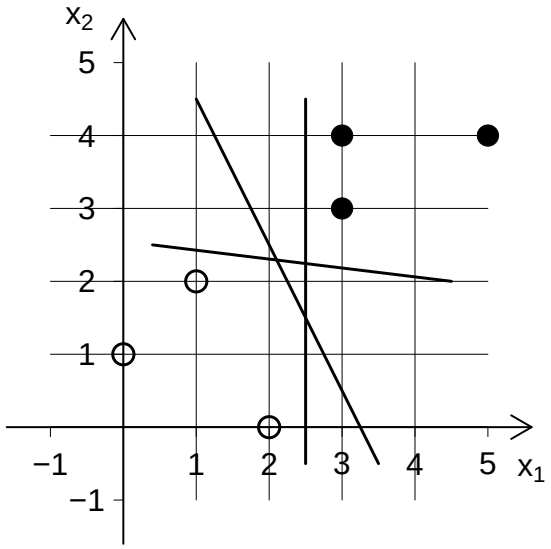


Figure 3: *Three lines that separate black and white points.*

The answer turns out to be the line

$$x_2 = 6.5 - 2x_1 \tag{1}$$

shown in Figure 4 along with the maximum margin rectangle, shown in grey. The *margin*, denoted by M, is the width of the rectangle and is equal to $\sqrt{5}$ in this case. We call (1) the *maximum margin line*. The maximum margin rectangle is "supported" on the 3 circled points. The points are known as *support vectors*, which is why the classification method being described in this article is so-named.

We now present the mathematics used to obtain this result. While the presentation is specific to these two-dimensional data, the extension to general training data and higher dimensions is fairly straightforward. We begin with:

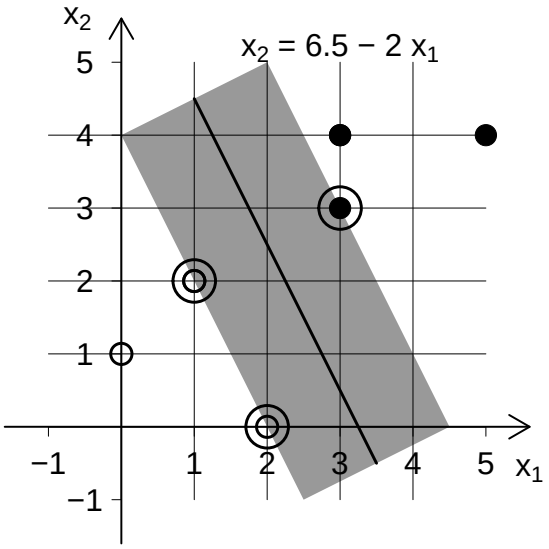


Figure 4: *The maximum margin line for the data of Figure 2. The circled points are the support vectors.*

Fundamental Result from Analytic Geometry
The *signed distance* from the point (x_1^*, x_2^*) to the line

$$\beta_0 + \beta_1 x_1 + \beta_2 x_2 = 0$$

is

$$\frac{\beta_0 + \beta_1 x_1^* + \beta_2 x_2^*}{\sqrt{\beta_1^2 + \beta_2^2}}.$$

Remark 1: The signed distance is such that points on opposite sides of the line have opposite signs. However, the actual signs depend on the choice of β_0, β_1 and β_2. If these β_i are multiplied through by a negative number then the positive distances become negative and vice versa.

■

Remark 2: Note that in the general line formulation

$$\beta_0 + \beta_1 x_1 + \beta_2 x_2 = 0$$

β_0, β_1 and β_2 are only defined up to a scalar multiplication. For example, the line

$$3 - 7x_1 + 4x_2 = 0$$

can also be expressed as

$$15 - 35x_1 + 20x_2 = 0$$

or

$$-6 + 14x_1 - 8x_2 = 0.$$

■

Let

$$\beta_0 + \beta_1 x_1 + \beta_2 x_2 = 0$$

be a general line separating the points in Figure 2 and let $M = M(\beta_0, \beta_1, \beta_2)$ denote the margin of the line, so that

$$M/2 = \text{half the margin.}$$

Label the points as follows:

$$\left.\begin{array}{l}(x_{11}, x_{12}) = (5, 4)\\ (x_{21}, x_{22}) = (3, 4)\\ (x_{31}, x_{32}) = (3, 3)\end{array}\right\} \text{ black points}$$

$$\left.\begin{array}{l}(x_{41}, x_{42}) = (1, 2)\\ (x_{51}, x_{52}) = (2, 0)\\ (x_{61}, x_{62}) = (0, 1)\end{array}\right\} \text{white points.}$$

We will suppose without loss of generality (courtesy of Remark 1) that the β_0, β_1 and β_2 are such that black points have negative signed distance to the line. Then from the above result:

$$\frac{\beta_0 + \beta_1\, x_{i1} + \beta_2\, x_{i2}}{\sqrt{\beta_1^2 + \beta_2^2}} \leq -\frac{M}{2}, \quad 1 \leq i \leq 3.$$

The white points must then have positive signed distance to the line and from the above result:

$$\frac{\beta_0 + \beta_1\, x_{i1} + \beta_2\, x_{i2}}{\sqrt{\beta_1^2 + \beta_2^2}} \geq \frac{M}{2}, \quad 4 \leq i \leq 6.$$

Let y_i, $1 \leq i \leq 6$, denote the class labels as follows:

$$y_i = \begin{cases} -1, & 1 \leq i \leq 3 \quad \text{(black points)}\\ 1, & 4 \leq i \leq 6 \quad \text{(white points).}\end{cases}$$

Then the previous two inequalities can be combined into the set of constraints

$$\frac{y_i(\beta_0 + \beta_1\, x_{i1} + \beta_2\, x_{i2})}{\sqrt{\beta_1^2 + \beta_2^2}} \geq \frac{M}{2}, \quad 1 \leq i \leq 6.$$

The problem is to maximise $M = M(\beta_0, \beta_1, \beta_2)$ subject to these constraints. Formally, we have the *constrained optimisation problem*:

$$\begin{aligned}&\max_{\beta_0,\beta_1,\beta_2} M = M(\beta_0, \beta_1, \beta_2)\\ &\text{subject to} \quad \frac{y_i(\beta_0 + \beta_1\, x_{i1} + \beta_2\, x_{i2})}{\sqrt{\beta_1^2 + \beta_2^2}} \geq \frac{M}{2} \text{ for all } 1 \leq i \leq 6.\end{aligned} \tag{2}$$

Since, as mentioned in Remark 2, the β_i's can be arbitrarily rescaled we can choose them to satisfy

$$\sqrt{\beta_1^2 + \beta_2^2} = 2/M.$$

Then $M = 2/\sqrt{\beta_1^2 + \beta_2^2}$ and maximising M is equivalent to minimising $\beta_1^2 + \beta_2^2$. So the constrained optimisation problem becomes

$$\begin{aligned}&\min_{\beta_0,\beta_1,\beta_2} \tfrac{1}{2}(\beta_1^2 + \beta_2^2)\\ &\text{subject to} \quad y_i(\beta_0 + \beta_1\, x_{i1} + \beta_2\, x_{i2}) \geq 1 \text{ for all } 1 \leq i \leq 6.\end{aligned}$$

(The inconsequential $\frac{1}{2}$ factor has been added as a multiplier to $\beta_1^2 + \beta_2^2$ to make the ensuing expressions look a bit nicer). Introduction of *Lagrange multipliers* $\alpha_1, \ldots, \alpha_6 \geq 0$ leads to the *(primal) Lagrangian*

$$L_P = \tfrac{1}{2}(\beta_1^2 + \beta_2^2) - \sum_{i=1}^{6} \alpha_i[y_i(\beta_0 + \beta_1\, x_{i1} + \beta_2\, x_{i2}) - 1].$$

Now

$$\begin{aligned}\frac{\partial L_P}{\partial \beta_0} &= -\sum_{i=1}^{6} \alpha_i y_i \\ \frac{\partial L_P}{\partial \beta_1} &= \beta_1 - \sum_{i=1}^{6} \alpha_i y_i x_{i1} \\ \frac{\partial L_P}{\partial \beta_2} &= \beta_2 - \sum_{i=1}^{6} \alpha_i y_i x_{i2}.\end{aligned}$$

Setting the first of these to zero leads to the constraint

$$\sum_{i=1}^{6} \alpha_i y_i = 0.$$

Setting the second and third of these to zero and substituting into the above primal Lagrangian leads to the dual Lagrangian

$$L_D = \sum_{i=1}^{6} \alpha_i - \tfrac{1}{2} \sum_{i=1}^{6} \sum_{j=1}^{6} \alpha_i \alpha_j y_i y_j (x_{i1} x_{j1} + x_{i2} x_{j2}).$$

The dual optimisation problem is then

$$\max_{\alpha_1,\ldots,\alpha_6} \textstyle\sum_{i=1}^{6} \alpha_i - \frac{1}{2} \sum_{i=1}^{6} \sum_{j=1}^{6} \alpha_i \alpha_j y_i y_j (x_{i1} x_{j1} + x_{i2} x_{j2})$$

$$\text{subject to } \alpha_i \geq 0, \text{ for all } 1 \leq i \leq 6, \text{ and } \textstyle\sum_{i=1}^{6} \alpha_i y_i = 0.$$

This is a *quadratic programming problem* in the α_i's (quadratic objective function and linear constraints) and there are now a number of routines available in standard computing packages (e.g. `Matlab` and `R`) to solve this. Let $\widehat{\alpha}_1, \ldots, \widehat{\alpha}_6$ denote the solutions. Then from the above partial derivatives being set to zero we get

$$\begin{aligned}\widehat{\beta}_1 &= \sum_{i=1}^{6} \widehat{\alpha}_i y_i x_{i1}, \\ \widehat{\beta}_2 &= \sum_{i=1}^{6} \widehat{\alpha}_i y_i x_{i2}\end{aligned}$$

and the corresponding maximum margin is

$$\widehat{M} = 2/\sqrt{\widehat{\beta}_1^2 + \widehat{\beta}_2^2}.$$

The intercept $\widehat{\beta}_0$ is simply the one that puts the line in the middle of the margin region.

For the data of Figure 2 quadratic programming software gives

$$\widehat{\alpha}_1 = \widehat{\alpha}_2 = \widehat{\alpha}_5 = \widehat{\alpha}_6 = 0 \quad \text{and} \quad \widehat{\alpha}_5 = \widehat{\alpha}_6 = 0.4.$$

The leads to

$$\widehat{\beta}_1 = 0.4 \times 1 \times 3 + 0.4 \times (-1) \times 1 = 0.8$$

and

$$\widehat{\beta}_2 = 0.4 \times 1 \times 3 + 0.4 \times (-1) \times 2 = 0.4$$

so the maximum margin line is of the form

$$\beta_0 + 0.8\, x_1 + 0.4\, x_2 = 0$$

or equivalently

$$\beta_0' + 2\,x_1 + \,x_2 = 0$$

where $\beta_0' = 2.5\beta_0$. The intercept that puts the line in the middle of the margin region is easily shown to be $\widehat{\beta}_0' = 6.5$ and we get the answer

$$6.5 + 2\,x_1 + \,x_2 = 0$$

as depicted in Figure 4 (or, if you don't like decimals, $13 + 4\,x_1 + 2\,x_2 = 0$). The corresponding maximum margin is

$$2/\sqrt{\widehat{\beta}_1^2 + \widehat{\beta}_2^2} = \sqrt{5} \simeq 2.236.$$

Typical quadratic programming software works with matrix formulation of the objective function. The general formulation is usually something like

$$\min_{\boldsymbol{\alpha}}(-\mathbf{d}^T\boldsymbol{\alpha} + \tfrac{1}{2}\boldsymbol{\alpha}^T\mathbf{D}\boldsymbol{\alpha})$$

where $\boldsymbol{\alpha}$ is the column vector containing the α_i, $\mathbf{d}$ is a column vector and $\mathbf{D}$ is a square matrix, each with the same number of rows as $\boldsymbol{\alpha}$. In our problem $\mathbf{d} = \mathbf{1}$ where $\mathbf{1}$ is a column vector with each entry equal to 1. Also,

$$\mathbf{D} = (\mathbf{y}\mathbf{y}^T) \odot (\mathbf{X}\mathbf{X}^T) \tag{3}$$

where

$$\mathbf{y} = \begin{bmatrix} y_1 \\ \vdots \\ y_6 \end{bmatrix}, \qquad \mathbf{X} = \begin{bmatrix} x_{11} & x_{12} \\ \vdots & \vdots \\ x_{61} & x_{62} \end{bmatrix}$$

and $\odot$ denotes element-wise product (i.e. the matrix you get when you multiply each of the corresponding entries together). This leads to the following matrix formulation of the quadratic programming problem:

$$\begin{aligned} &\min_{\boldsymbol{\alpha}}(-\mathbf{1}^T\boldsymbol{\alpha} + \tfrac{1}{2}\boldsymbol{\alpha}^T\mathbf{D}\boldsymbol{\alpha}) \\ &\text{subject to } \alpha_i \geq 0, \text{ for all } 1 \leq i \leq 6, \text{ and } \textstyle\sum_{i=1}^6 \alpha_i y_i = 0. \end{aligned} \tag{4}$$

In practice is more common that the black and white points cannot be separated by a straight line. The data depicted in Figure 5 is of this type. Figure 6 shows an example line and corresponding margin region for the data of Figure 5. The dotted lines in Figure 6 correspond to the the non-zero values of

$$\begin{aligned} \xi_i \;\; = \;\; &\text{extent to which the } i\text{th point is 'on the wrong side'} \\ &\text{of the margin region boundary as a proportion of } M. \end{aligned}$$

The maximum margin problem in this non-separable case is then of the form:

$$\begin{aligned} &\max_{\beta_i, \xi_i} M = M(\beta_0, \beta_1, \beta_2, \xi_1, \ldots, \xi_6) \\ &\text{subject to} \quad \frac{y_i(\beta_0 + \beta_1\,x_{i1} + \beta_2\,x_{i2})}{\sqrt{\beta_1^2 + \beta_2^2}} \geq \frac{M(1 - \xi_i)}{2}, \\ &\qquad\qquad \xi_i \geq 0 \;\text{ for all }\; 1 \leq i \leq 6 \quad \text{and} \quad \textstyle\sum_{i=1}^6 \xi_i < C. \end{aligned} \tag{5}$$

Here $C > 0$ is a tuning parameter. It sometimes referred to as a *cost* parameter since it controls the degree of violation ('cost') of separability by the maximum margin line.

Introducing Lagrange multipliers applying mathematical arguments similar to those given in the separable case given on page 5 leads to the following generalisation of (4):

$$\min_{\boldsymbol{\alpha}}(-\mathbf{1}^T\boldsymbol{\alpha} + \tfrac{1}{2}\boldsymbol{\alpha}^T\mathbf{D}\boldsymbol{\alpha})$$
$$\text{subject to } 0 \leq \alpha_i \leq \tfrac{1}{2}C, \text{ for all } 1 \leq i \leq n, \text{ and } \textstyle\sum_{i=1}^{n} \alpha_i y_i = 0. \tag{6}$$

The only difference between (4) and (6) is that the former has $C = \infty$, whereas the latter has a finite C. In closing, it should be noted that while these notes have just treated two specific problems with $n = 6$ the extension to general n is quite straightforward. Similarly, the extension to higher dimensions and maximum margin hyperplanes involves algebra not much more involved than that given here. This results in what are now called *linear support vector machine* classifiers.

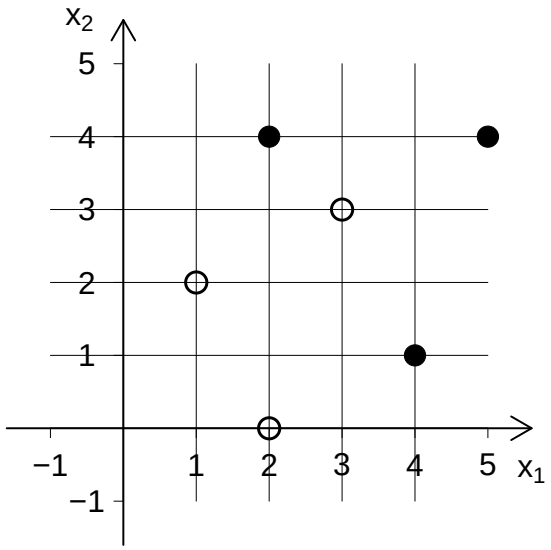


Figure 5: *An example two-dimensional training data set where the points are not separable by a straight line.*

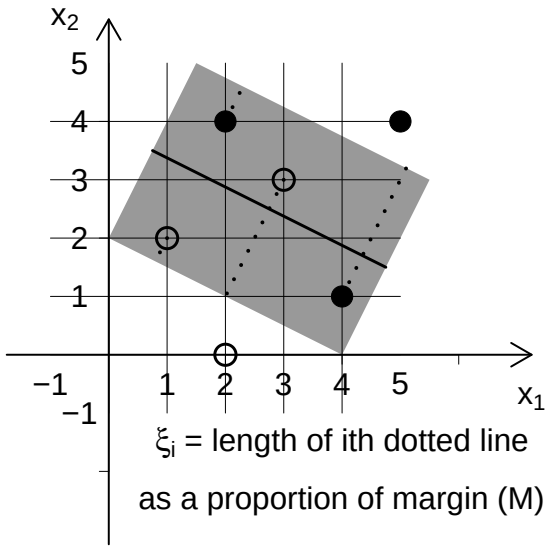


Figure 6: *An example line and corresponding margin with the ξ_i's denoting the extent to which some of the points are 'on the wrong side' of the margin region boundary.*

Kernelisation. Now consider the points plotted in Figure 7. Clearly the black and white points are not well-separated by any straight line. However, as shown in Figure 8, they

can be separated by an ellipse of the form:

$$\frac{x_1^2}{a^2} + \frac{x_2^2}{b^2} = 1. \tag{7}$$

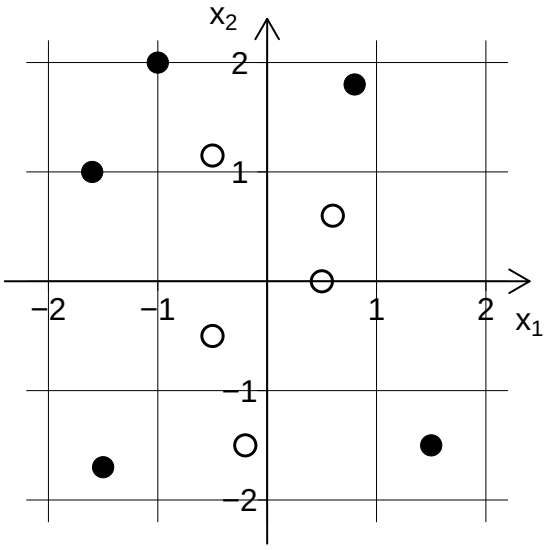


Figure 7: *A training data set which is not naturally separable by a straight line.*

Introduce the new variables z_1 and z_2:

$$z_1 = x_1^2 \qquad \text{and} \qquad z_2 = x_2^2.$$

Then (7) becomes

$$\frac{z_1}{a^2} + \frac{z_2}{b^2} = 1.$$

which corresponds to the family of (negatively sloped) straight lines in the (z_1, z_2) coordinate space. Figure 9 shows the points in this transformed space and separability by straight lines is apparent. If we now apply the maximum margin line methodology from the previous section to these data then we obtain the result shown in Figure 10.

Transforming back to the original space leads to the ellipsoidal separator shown in Figure 11. Notice that the margin region becomes an 'elastic band'-like region in the (x_1, x_2) space.

The process by which the points are transformed to a new coordinate system, have a maximum margin line (or hyperplane) fitted to them, which is then transformed back to the original space allows for complex non-linear separating boundaries. It is known as *kernelisation* and the remainder of this section will be concerned with explaining the reason for this name.

First of all note from (3) and (4) that the maximum margin line algorithm depends on the x_{1i} and x_{2i} only through the entries of the matrix

$$\mathbf{X}\mathbf{X}^T = [\mathbf{x}_i^T \mathbf{x}_j]_{1 \le i,j \le n}$$

where $\mathbf{x}_i = [x_{1i}\, x_{2i}]^T$. So the maximum margin line for the z_1, z_2 data (Figure 10) depends only on the z_{1i} and z_{2i} through pairwise products of the form

$$\mathbf{z}_i^T \mathbf{z}_j = z_{1i}z_{1j} + z_{2i}z_{2j} = x_{1i}^2 x_{1j}^2 + x_{2i}^2 x_{2j}^2 = K(\mathbf{x}_i, \mathbf{x}_j)$$

where, for general pairs $\mathbf{s} = (s_1, s_2)$ and $\mathbf{t} = (t_1, t_2)$,

$$K(\mathbf{s}, \mathbf{t}) = s_1^2 t_1^2 + s_2^2 t_2^2.$$

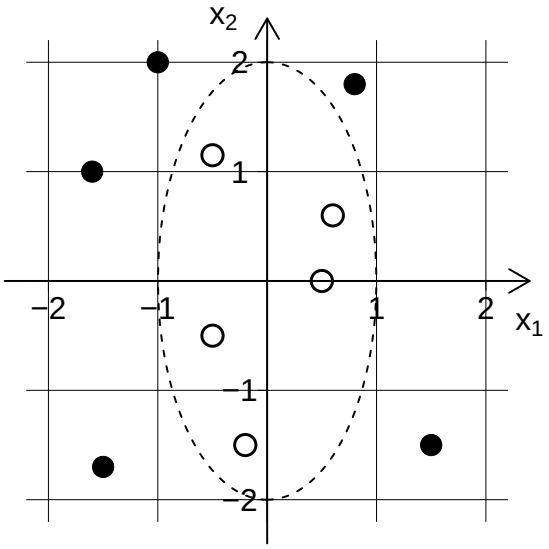


Figure 8: *An ellipse that separates the black and white points.*

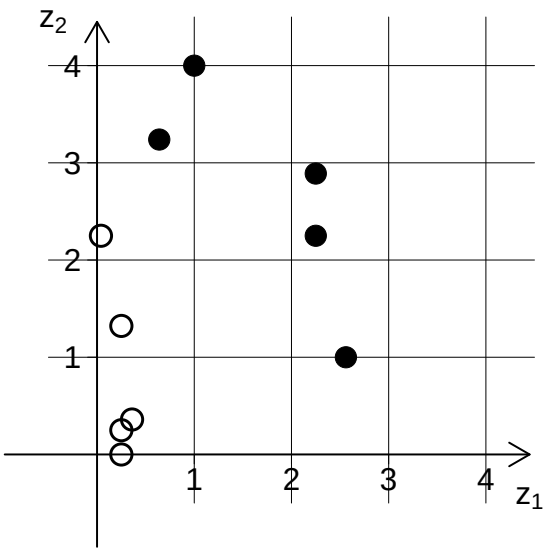


Figure 9: *The training data in the* (z_1, z_2) *space.*

The bivariate function K is known as a *kernel* function and characterises the transformation from the (x_1, x_2) space to the (z_1, z_2) space. However, note that the maximum margin line algorithm depends only on the matrix

$$[K(\mathbf{x}_i, \mathbf{x}_j)]_{1 \leq i,j \leq n}.$$

We do not have to explicitly work with the (z_{1i}, z_{2i}) values. This opens up the possibility of using other bivariate functions with the idea that they will allow more complex separating boundaries. Some kernel functions that are commonly used in practice are:

Radial basis:	$K(\mathbf{s}, \mathbf{t}) = \exp\{-\gamma(\mathbf{s} - \mathbf{t})^T(\mathbf{s} - \mathbf{t})\}$
pth degree polynomial:	$K(\mathbf{s}, \mathbf{t}) = (1 + \mathbf{s}\mathbf{t}^T)^p$
tanh:	$K(\mathbf{s}, \mathbf{t}) = \tanh(\kappa_1 \mathbf{s}\mathbf{t}^T + \kappa_2)$

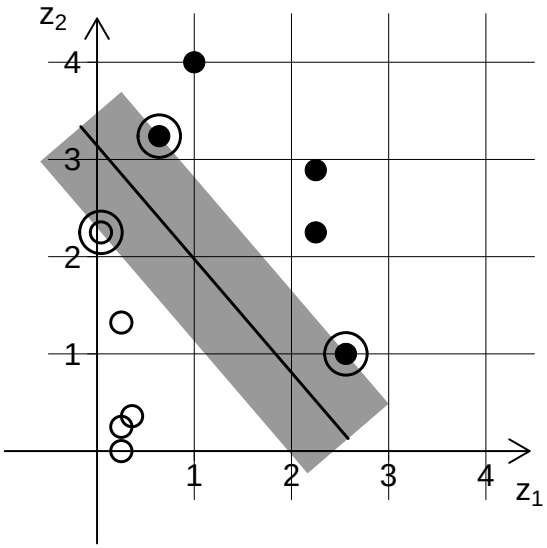


Figure 10: *The maximum margin line for the training data in the* (z_1, z_2) *space. The circled points are the support vectors.*

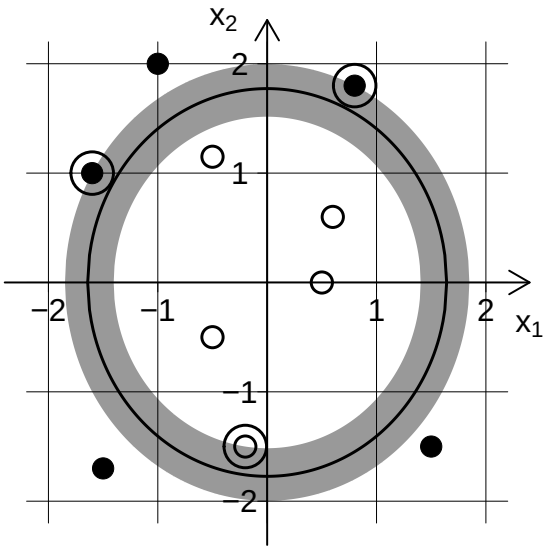


Figure 11: *The classifier and margin region from Figure 10 when transformed back to the* (x_1, x_2) *space. The circled points are the support vectors.*

The radial basis kernel is the most popular. It corresponds to transforming from (x_1, x_2) to an 'infinite dimensional' space.

We are now ready to give the full *Support Vector Machine Classification Algorithm*:

Support Vector Machine Classification Algorithm
Inputs: $\mathbf{x}_i \in \mathbb{R}^d$, $y_i \in \{-1, 1\}$, $1 \leq i \leq n$.

1. Choose a *kernel* $K(\mathbf{s}, \mathbf{t})$.
2. Choose a *cost parameter* C.
3. Form the *Gram matrix* $\mathbf{K} = [K(\mathbf{x}_i, \mathbf{x}_j)]_{1 \leq i,j \leq n}$.
4. Form the matrix $\mathbf{D} = (\mathbf{y}\mathbf{y}^T) \odot \mathbf{K}$ and solve the quadratic programming problem

$$\begin{aligned} &\min_{\boldsymbol{\alpha}}(-\mathbf{1}^T\boldsymbol{\alpha} + \tfrac{1}{2}\boldsymbol{\alpha}^T\mathbf{D}\boldsymbol{\alpha}) \\ &\text{subject to } 0 \leq \alpha_i \leq \tfrac{1}{2}C, \text{ for all } 1 \leq i \leq n, \text{ and } \textstyle\sum_{i=1}^n \alpha_i y_i = 0. \end{aligned} \tag{8}$$

5. Get an intercept estimate $\widehat{\beta}_0$ using *Karush-Kuhn-Tucker constraints* (not explained here, but easy).
6. The

$$\widehat{f}(\mathbf{x}) = \widehat{\beta}_0 + \sum_{i=1}^{n} \widehat{\alpha}_i K(\mathbf{x}, \mathbf{x}_i) y_i$$

is then used to a classify future feature measurement $\mathbf{x}$, based on the sign of $\widehat{f}(\mathbf{x})$.

Computer-aided Mail Sorting. We now return to the problem of computer-aided mail sorting. Note, again, the handwritten digits in Figure 1. How can we teach a computer to distinguish between a handwritten '5' and a handwritten '6', say? The first step is to turn a handwritten digit into a numerical object. This can done by scanning the digit into a computer and obtaining a *grey level* representation. The left panel of Figure 12 shows such a representation. The scanning process has produced a 16×16 array of *pixels* of shades of grey. The next step is to convert the shades of grey to numbers. A common convention is to use the integers 0,1,...,255 where 0=white and 255=black. For illustration purposes we will use the simpler scale: 0=white and 9=black This leads to the 16×16 array of numbers shown in the right panel of Figure 12.

The *vector* representation of the handwritten '5' is then:

$$[0\,0\,0\,\cdots 0\,4\,3\,7\,7\,9\,8\,7\,8\,7\,5\,0\,0\,\cdots 0\,0\,0]^T.$$

which is in the space $\mathbb{R}^{256}$. It has been reported in the literature that ninth degree polynomial kernels have lead to good classification performance for handwritten digits. This kernel is given by

$$K(\mathbf{s}, \mathbf{t}) = (1 + \mathbf{s}^T\mathbf{t})^9.$$

If

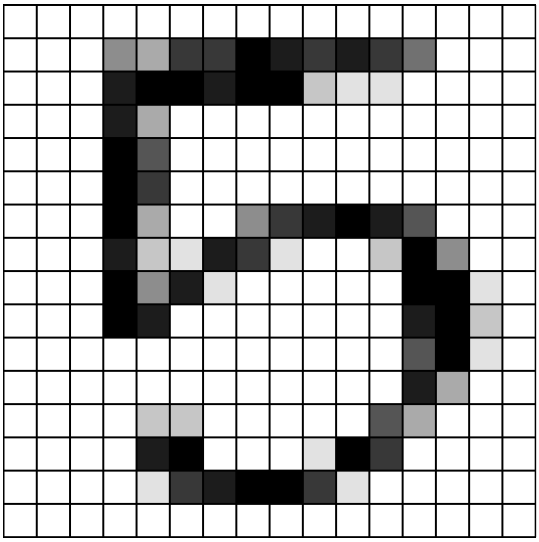
$$\mathbf{x}_1 = [x_{11}\ x_{12}\ \cdots\ x_{1,256}]^T.$$

is a digital version of a handwritten '5' then $y_1 = -1$. If

$$\mathbf{x}_2 = [x_{21}\ x_{12}\ \cdots\ x_{2,256}]^T$$

is a digital version of a handwritten '6' then $y_2 = 1$. Then the support vector machine requires

$$K(\mathbf{x}_1, \mathbf{x}_2) = (1 + \mathbf{x}_1^T\mathbf{x}_2)^9 = \left(1 + \sum_{\ell=1}^{256} x_{1\ell}x_{2\ell}\right)^9.$$



0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
0	0	0	4	3	7	7	9	8	7	8	7	5	0	0	0
0	0	0	8	9	9	8	9	9	2	1	1	0	0	0	0
0	0	0	8	3	0	0	0	0	0	0	0	0	0	0	0
0	0	0	9	6	0	0	0	0	0	0	0	0	0	0	0
0	0	0	9	7	0	0	0	0	0	0	0	0	0	0	0
0	0	0	9	3	0	0	4	7	8	9	8	6	0	0	0
0	0	0	8	2	1	8	7	1	0	0	2	9	4	0	0
0	0	0	9	4	8	1	0	0	0	0	0	9	9	1	0
0	0	0	9	8	0	0	0	0	0	0	0	8	9	2	0
0	0	0	0	0	0	0	0	0	0	0	0	6	9	1	0
0	0	0	0	0	0	0	0	0	0	0	0	8	3	0	0
0	0	0	0	2	2	0	0	0	0	0	6	3	0	0	0
0	0	0	0	8	9	0	0	0	1	9	7	0	0	0	0
0	0	0	0	1	7	8	9	9	7	1	0	0	0	0	0
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Figure 12: *Grey level representation of a handwritten '5', over a 16×16 pixel array and its corresponding numerical (or digital) representation.*

Suppose that we have available 1000 such pairs:

$$(\mathbf{x}_i, y_i), \quad 1 \leq i \leq 1000.$$

Then we repeat the above calculation to obtain

$$K(\mathbf{x}_i, \mathbf{x}_j), \quad 1 \leq i, j \leq 1000.$$

These values fill up the *Gram* matrix $\mathbf{K}$ of the Support Vector Machine Classification Algorithm given above. A computer can classify future images as a '5' or '6' (although not necessarily perfectly) via this algorithm.

Further reading

If you would like to read further on classification methods, and the role played by mathematics, then a recommended book is *The Elements of Statistical Learning* by Trevor Hastie, Robert Tibshirani & Jerome Friedman (2001, Springer-Verlag). All three authors are Statistics professors at Stanford University in California, USA, and they have gone to considerable trouble to make this area of research accessible to a wider audience.