Maximum margin classifiers

Sparse kernel methods

Francesco Corona

Outline

Maximum margin classifiers
Sparse kernel methods

Maximum margin classifiers

We explored a variety of learning algorithms based on nonlinear kernels

One of the significant limitations of many such algorithms:

- The kernel function $k(\mathbf{x}_n, \mathbf{x}_m)$ must be evaluated for all possible pairs $\mathbf{x}_n$ and $\mathbf{x}_m$ of training points
- Computationally infeasible during training and can lead to excessive computation times when making predictions for new data points

We now discuss at kernel-based algorithms that have sparse solutions

- Predictions for new inputs will depend only on the kernel function evaluated at a subset of the training data points

We look in some detail at the **Support vector machine** (SVM)

- An important property: The determination of the model parameters corresponds to a convex optimisation problem
- Any local solution is also a global optimum

Maximum margin classifiers (cont.)

We begin with a two-class classification problem using linear models of the form

$$y(\mathbf{x}) = \mathbf{w}^T \phi(\mathbf{x}) + b \tag{1}$$

$\phi(\mathbf{x})$ is a fixed feature-space transformation and the bias parameter b is explicit

- The training data set comprises N input vectors $\mathbf{x}_1, , \ldots, \mathbf{x}_N$
- Target values $t_1, , \ldots, t_N$, where $t_n \in \{-1, 1\}$
- New data points $\mathbf{x}$ are classified based on the sign of $y(\mathbf{x})$

We assume first that the training data set is linearly separable in feature space

- By definition, there exists at least one choice of parameters $\mathbf{w}$ and b such that a function $y(\mathbf{x})$
- $y(\mathbf{x}_n) > 0$ for points having $t_n = +1$
- $y(\mathbf{x}_n) < 0$ for points having $t_n = -1$

In summary, $t_n y(\mathbf{x}_n) > 0$ for all N training points

Maximum margin classifiers (cont.)

There may of course exist many such solutions that separate the classes exactly

The perceptron is guaranteed to find a solution in a finite number of steps

- The solution that it finds is dependent on the (arbitrary) initial values for $\mathbf{w}$ and b as well as on the order in which data points are presented

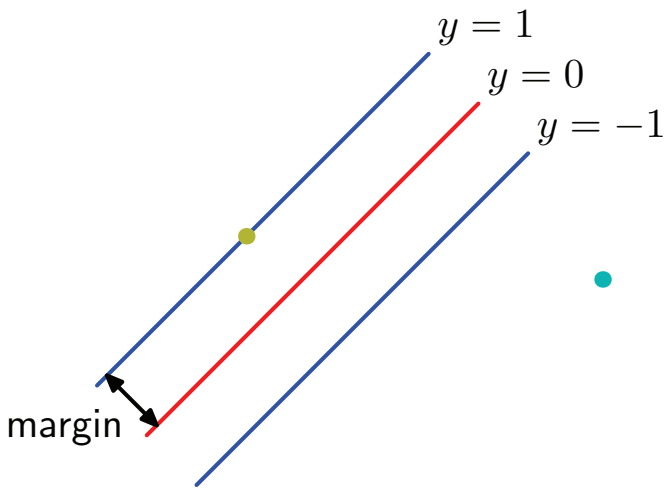
If there are multiple solutions all of which classify the training data set exactly, then we should try to find the one that will give the smallest generalisation error

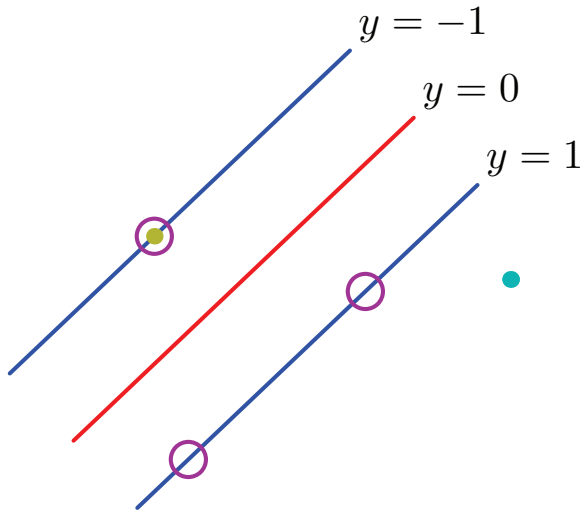
- The SVM approaches this problem through the concept of **margin**
- The smallest distance between decision boundary and any of the samples

Maximum margin classifiers (cont.)

The margin is defined as the perpendicular distance between the decision boundary and the closest of the data points (left figure)

Maximizing the margin leads to a choice of decision boundary (right figure)



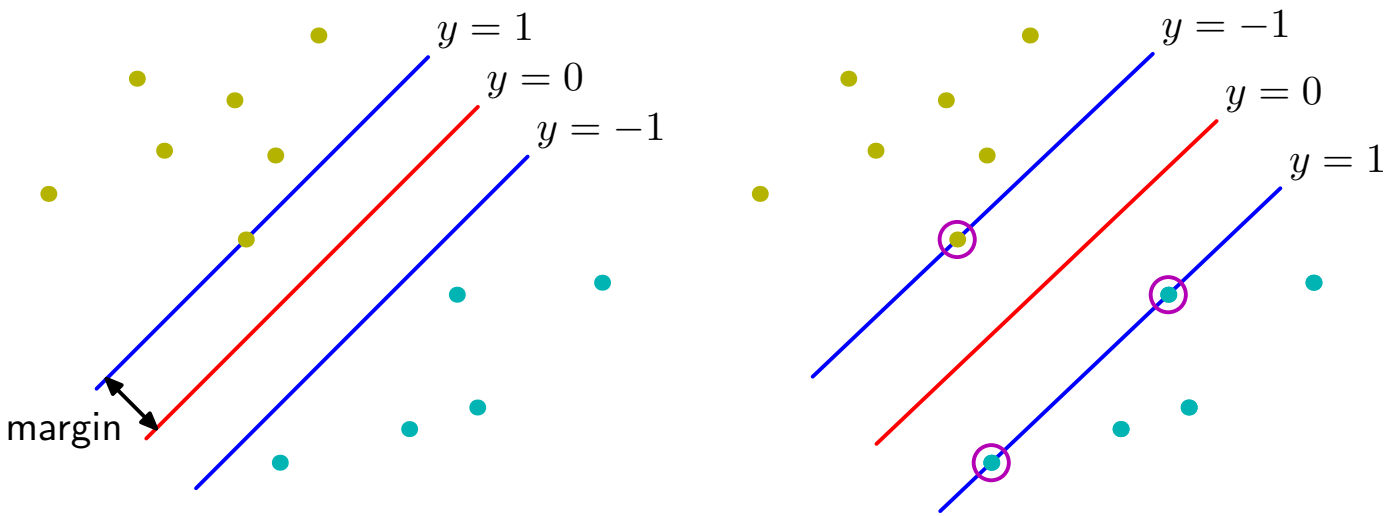


The location of this boundary is determined by a subset of the data points, known as **support vectors**, which are indicated by the circles

Maximum margin classifiers (cont.)

The margin is defined as the perpendicular distance between the decision boundary and the closest of the data points (left figure)

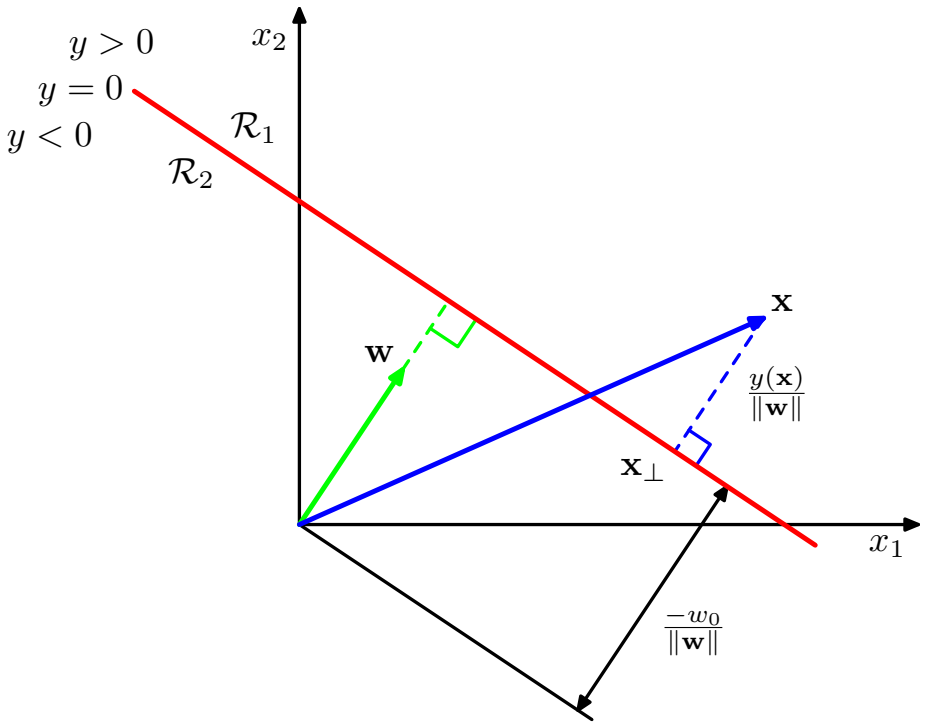
Maximizing the margin leads to a choice of decision boundary (right figure)



The location of this boundary is determined by a subset of the data points, known as **support vectors**, which are indicated by the circles

Maximum margin classifiers (cont.)

Recall that the perpendicular distance of a point $\mathbf{x}$ from the hyperplane $y(\mathbf{x}) = 0$ where $y(\mathbf{x}) = \mathbf{w}^T\phi(\mathbf{x}) + b$ equals $\dfrac{|y(\mathbf{x})|}{||\mathbf{w}||}$



We are only interested in solutions for which all data points are correctly classified

$$t_n y(\mathbf{x}_n) > 0, \quad \text{for all } n$$

The distance of a point $\mathbf{x}_n$ to the decision surface is given by

$$\frac{t_n y(\mathbf{x}_n)}{||\mathbf{w}||} = \frac{t_n(\mathbf{w}^T\phi(\mathbf{x}_n) + b)}{||\mathbf{w}||} \tag{2}$$

The margin is given by the perpendicular distance to the closest point $\mathbf{x}_n$

Maximum margin classifiers (cont.)

$$\frac{t_n y(\mathbf{x}_n)}{||\mathbf{w}||} = \frac{t_n(\mathbf{w}^T\phi(\mathbf{x}_n) + b)}{||\mathbf{w}||}$$

We wish to optimise parameters $\mathbf{w}$ and b in order to maximise this distance

Thus the maximum margin solution is found by solving

$$\arg\max_{\mathbf{w},b} \left(\frac{1}{||\mathbf{w}||} \min_n \left(t_n(\mathbf{w}^T\phi(\mathbf{x}_n) + b) \right) \right) \tag{3}$$

where we have taken the factor $1/||\mathbf{w}||$ outside the optimisation over n because $\mathbf{w}$ does not depend on n

Direct solution of this optimisation problem would be very complex, and so we shall convert it into an equivalent problem that is much easier to solve

Maximum margin classifiers (cont.)

To do this we note that if we rescale $\mathbf{w} \rightarrow \kappa\mathbf{w}$ and $b \rightarrow \kappa b$, then the distance from any point $\mathbf{x}_n$ to the decision surface, given by $\frac{t_n y(\mathbf{x}_n)}{||\mathbf{w}||}$, remains unchanged

We will use this freedom to set

$$t_n(\mathbf{w}^T\phi(\mathbf{x}_n) + b) = 1 \tag{4}$$

for the point closest to surface

Maximum margin classifiers (cont.)

In this case, all data points will satisfy the **constraints**

$$t_n(\mathbf{w}^T\phi(\mathbf{x}_n) + b) \geq 1, \quad \text{with } n = 1, \ldots, N \tag{5}$$

the canonical representation of the decision hyperplane

- Points for which the equality holds, the constraints are said to be **active**
- Points for which the equality does not hold they are said to be **inactive**

By definition, there will always be (at least) one active constraint

- There will always be a closest point

Once the margin is maximised there will be two active constraints

Maximum margin classifiers (cont.)

The optimisation problem then simply requires that we maximise $\frac{1}{||\mathbf{w}||}$

- This is equivalent to minimising $||\mathbf{w}||^2$

We solve the optimisation problem

$$\arg\min_{\mathbf{w},b} \frac{1}{2}||\mathbf{w}||^2 \tag{6}$$

subjected to the given constraints

$$t_n(\mathbf{w}^T\phi(\mathbf{x}_n) + b) \geq 1, \text{ with } n = 1, \ldots, N$$

The factor of $1/2$ is only included for later convenience

Maximum margin classifiers (cont.)

$$\begin{aligned} \arg\min_{\mathbf{w},b} \quad & \frac{1}{2}||\mathbf{w}||^2 \\ \text{s.t.} \quad & t_n(\mathbf{w}^T\phi(\mathbf{x}_n) + b) \geq 1, \text{ with } n = 1, \ldots, N \end{aligned}$$

This is an example of a quadratic programming problem in which we are trying to minimise a quadratic function subject to a set of linear inequality constraints

It appears that the bias parameter b has disappeared from the optimisation

- However, it is determined implicitly via the constraints, as these require that changes to $||\mathbf{w}||$ be compensated by changes to b

Maximum margin classifiers (cont.)

To solve the constrained optimisation, we use Lagrange multipliers $a_n \geq 0$

- One multiplier a_n for each one of the N constraints

The resulting Lagrangian function takes the form

$$L(\mathbf{w}, b, \mathbf{a}) = \frac{1}{2}||\mathbf{w}||^2 - \sum_{n=1}^{N} a_n \Big(t_n(\mathbf{w}^T \phi(\mathbf{x}_n) + b) - 1 \Big) \tag{7}$$

where we have defined the vector $\mathbf{a} = (a_1, \ldots, a_n)^T$

Lagrange multipliers

Lagrange multipliers are used to find the stationary points of a function of several variables subject to one or more constraints

Consider the problem of finding the maximum of a function $f(x_1, x_2)$ subject to a constraint $g(x_1, x_2) = 0$ that relates variables x_1 and x_2

1. One could solve the constraint equation $g(x_1, x_2) = 0$ and thus express x_2 as a function of x_1 in the form $x_2 = h(x_1)$
2. This can be substituted into $f(x_1, x_2)$ to give a function of x_1 alone of the form $f(x_1, h(x_1))$
3. The maximum with respect to x_1 could be found by differentiation
4. To give a stationary value x_1^* and corresponding value $x_2^* = h(x_1^*)$

Lagrange multipliers (cont.)

However, it may be quite difficult to find an analytic solution of the constraint equation that allows x_2 (x_1) to be expressed as an explicit function of x_1 (x_2)

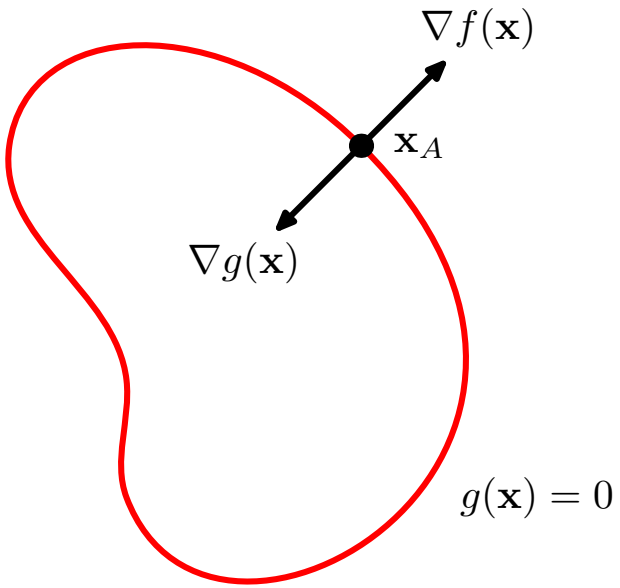
- Also, this approach treats x_1 and x_2 differently

A more elegant, and often simpler, approach is based on the introduction of a parameter λ called a **Lagrange multiplier**

We motivate this technique from a geometrical perspective

Lagrange multipliers (cont.)

Consider a D-dimensional variable $\mathbf{x}$ with components $x_1, \ldots, x_D$

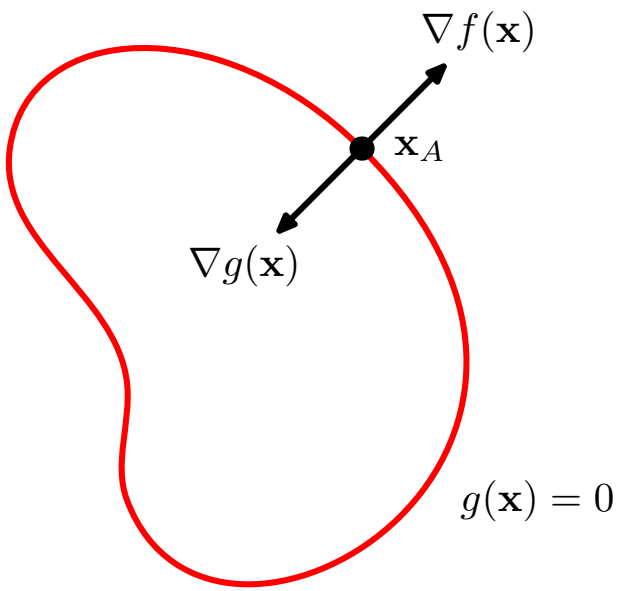


The constraint equation $g(\mathbf{x}) = 0$ represents a $(D-1)$-dimensional surface in the $\mathbf{x}$-space (red curve)

At any point on the constraint surface, the gradient $\nabla g(\mathbf{x})$ of the constraint function is orthogonal to the surface itself

Lagrange multipliers (cont.)

Consider a point $\mathbf{x}$ and point $\mathbf{x} + \varepsilon$ both on the constraint surface $g(\mathbf{x}) = 0$



Point $\mathbf{x}$ and point $\mathbf{x} + \varepsilon$ are closeby

- Taylor expansion of $g(\mathbf{x})$ around $\mathbf{x}$

$$g(\mathbf{x} + \varepsilon) \simeq g(\mathbf{x}) + \varepsilon^T \nabla g(\mathbf{x})$$

Points $\mathbf{x}$ and $\mathbf{x} + \varepsilon$ are on $g(\mathbf{x}) = 0$

- $g(\mathbf{x}) = g(\mathbf{x} + \varepsilon) = 0$
- Thus, $\varepsilon^T \nabla g(\mathbf{x}) \simeq 0$

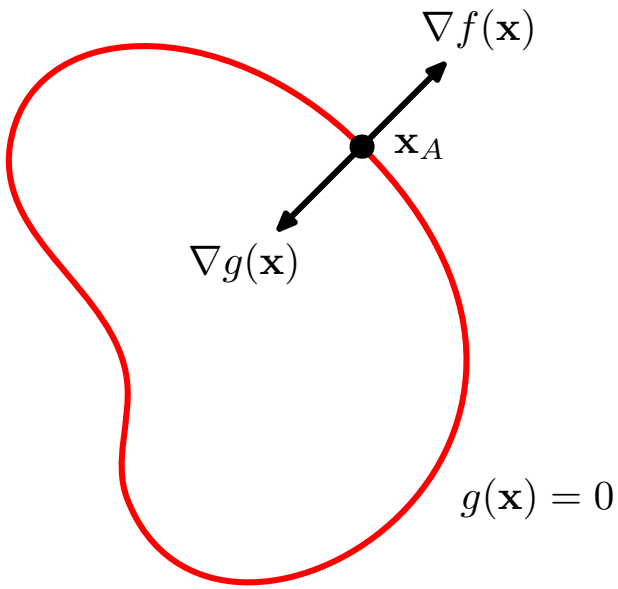
In the limit $||\varepsilon|| \to 0$, $\varepsilon^T \nabla g(\mathbf{x}) = 0$

Because ε is parallel to the constraint surface and because $\varepsilon^T \nabla g(\mathbf{x}) = 0$

- it follows that $\nabla g(\mathbf{x})$ must be normal to the surface

Lagrange multipliers (cont.)

We seek a point $\mathbf{x}^*$ on the constraint surface such that $f(\mathbf{x})$ is maximised



Such a point is such that vector $\nabla f(\mathbf{x})$ is also orthogonal to the constraint surface

- Otherwise, moving a short distance along the constraint would increase the value of $f(\mathbf{x})$

Therefore ∇g and ∇f are (anti-)parallel and there exists a parameter λ such that

$$\nabla f + \lambda \nabla g = 0$$

$\lambda \neq 0$ is known as a **Lagrange multiplier**

Lagrange multipliers (cont.)

It is convenient to define at this point the **Lagrangian function** given by

$$L(\mathbf{x}, \lambda) \equiv f(\mathbf{x}) + \lambda g(\mathbf{x})$$

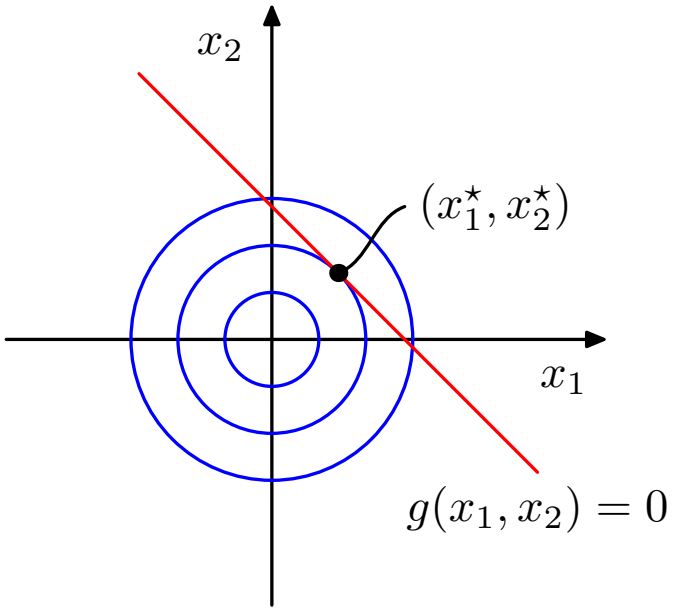
The constrained stationary condition $\nabla f + \lambda \nabla g = 0$ arises from $\nabla_{\mathbf{x}} L = 0$

In addition, condition $\dfrac{\partial L}{\partial \lambda} = 0$ leads to the constraint equation $g(\mathbf{x}) = 0$

To find the maximum of $f(\mathbf{x})$ subject to the constraint $g(\mathbf{x})$ we need to find the stationary points of $L(\mathbf{x}, \lambda)$ with respect to both $\mathbf{x}$ and λ

- For a D-dimensional variable $\mathbf{x}$, this gives $D + 1$ equations
- If uninteresting, the equation in λ can be eliminated

Lagrange multipliers (cont.)



Find the stationary point of function

$$f(x_1, x_2) = 1 - x_1^2 - x_2^2$$

subjected to the constraint function

$$g(x_1, x_2) = x_1 + x_2 - 1 = 0$$

The associated Lagrangian function is

$$L(\mathbf{x}, \lambda) = 1 - x_1^2 - x_2^2 + \lambda(x_1 + x_2 - 1)$$

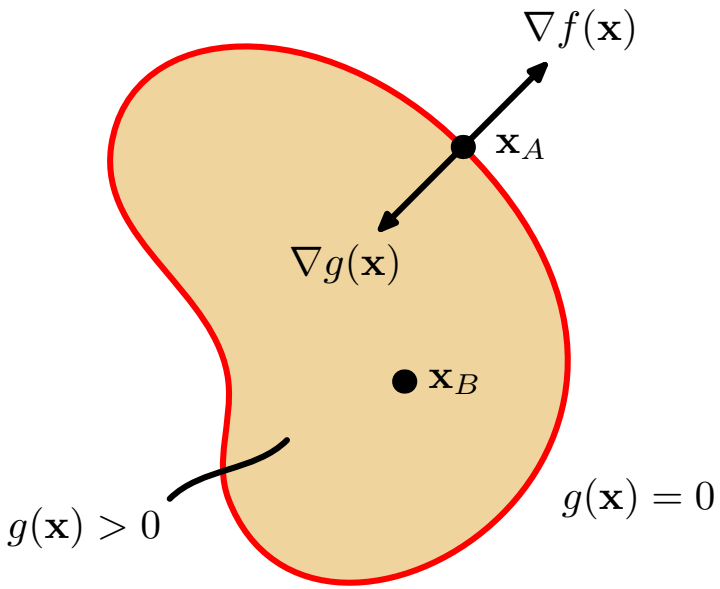
Conditions for the Lagrangian function to be stationary wrt x_1, x_2 and λ are

$$\begin{aligned} -2x_1 + \lambda &= 0 \\ -2x_2 + \lambda &= 0 \\ x_1 + x_2 - 1 &= 0 \end{aligned} \quad \Longrightarrow \quad (x_1^*, x_2^*) = \left(\frac{1}{2}, \frac{1}{2}\right) \quad \text{and} \quad \lambda = 1$$

Lagrange multipliers (cont.)

We consider the problem of maximising a function $f(\mathbf{x})$ subject to an inequality constraint of the form $g(\mathbf{x}) \geq 0$

There are two kinds of solution, depending on where the stationary points lies



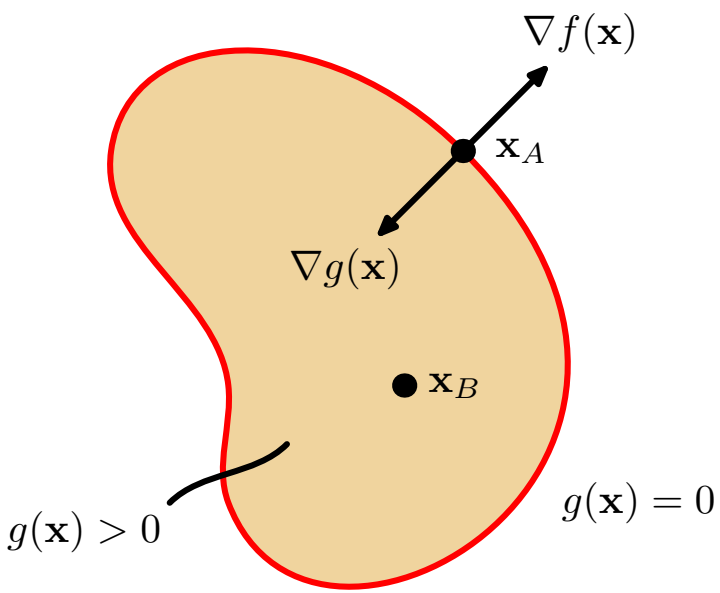
- The constrained stationary point lies in the region where $g(\mathbf{x}) > 0$, in which case the constraint is inactive
- The constrained stationary point lies on the boundary where $g(\mathbf{x}) = 0$, in which case the constraint is active

In the first case, the function $g(\mathbf{x})$ plays no role and so the stationary condition is simply $\nabla f(\mathbf{x}) = 0$, the stationary point of $L(\mathbf{x}, \lambda) = f(\mathbf{x}) + \lambda g(\mathbf{x})$ with $\lambda = 0$

Lagrange multipliers (cont.)

The second case is analogous to the case with an equality constraint $g(\mathbf{x}) = 0$

$$L(\mathbf{x}, \lambda) = f(\mathbf{x}) + \lambda g(\mathbf{x})$$



With a difference, it is now crucial the sign of the Lagrange multiplier

- Function $f(\mathbf{x})$ is at a maximum only if its gradient is oriented away from

$$g(\mathbf{x}) > 0$$

- $\nabla f(\mathbf{x}) = -\lambda \nabla g(\mathbf{x})$, for some $\lambda > 0$

Lagrange multipliers (cont.)

Whatever the stationary point is in or on the constraint region, $\lambda g(\mathbf{x}) = 0$

The solution to the maximisation of $f(\mathbf{x})$ subject to $g(\mathbf{x}) \geq 0$ is obtained by optimising the Lagrange function $L(\mathbf{x}, \lambda) = f(\mathbf{x}) + \lambda g(\mathbf{x})$ wrt $\mathbf{x}$ and λ

- subjected to the conditions we have just derived

$$\begin{aligned} g(\mathbf{x}) &\geq 0 \\ \lambda &\geq 0 \\ \lambda g(\mathbf{x}) &= 0 \end{aligned}$$

- The **Karush-Kuhn-Tucker** (KKT) conditions

If we wish to minimise (rather than maximise) function $f(\mathbf{x})$ subject to an inequality constraint $g(\mathbf{x}) \geq 0$, then we minimise the Lagrangian function

$$L(\mathbf{x}, \lambda) = f(\mathbf{x}) - \lambda g(\mathbf{x})$$

with respect to $\mathbf{x}$, again subject to $\lambda \geq 0$

Lagrange multipliers (cont.)

The technique of Lagrange multipliers can be extended to the case of multiple equality and inequality constraints

Consider the problem of finding the maximum of function $f(\mathbf{x})$ subject to constraints $g_j(\mathbf{x}) = 0$ for $j = 1, \ldots, J$ and $h_k(\mathbf{x}) \geq 0$ for $k = 1, \ldots, K$

We introduce Lagrange multipliers $\{\lambda_j\}$ and $\{\mu_k\}$, and then optimise the Lagrangian function

$$L(\mathbf{x}, \boldsymbol{\lambda}, \boldsymbol{\mu}) = f(\mathbf{x}) + \sum_{j=1}^{J} \lambda_j g_j(\mathbf{x}) + \sum_{k=1}^{K} \mu_k h_k(\mathbf{x})$$

subject to $\mu_k \geq 0$ and $\mu h_k(\mathbf{x}) = 0$ for $k = 1, \ldots, K$

Maximum margin classifiers (cont.)

$$L(\mathbf{w}, b, \mathbf{a}) = \frac{1}{2}||\mathbf{w}||^2 - \sum_{n=1}^{N} a_n \Big(t_n(\mathbf{w}^T \phi(\mathbf{x}_n) + b) - 1 \Big)$$

The minus sign in front of the Lagrange multipliers indicates that we are minimising with respect to $\mathbf{w}$ and b, and maximising with respect to $\mathbf{a}$

Setting the derivative of $L(\mathbf{w}, b, \mathbf{a})$ with respect to $\mathbf{w}$ and b equal to zero

$$\mathbf{w} = \sum_{n=1}^{N} a_n t_n \phi(\mathbf{x}_n) \tag{8}$$

$$0 = \sum_{n=1}^{N} a_n t_n \tag{9}$$

Maximum margin classifiers (cont.)

Eliminating $\mathbf{w}$ and b from $L(\mathbf{w}, b, \mathbf{a})$, using these two conditions, gives the dual representation of the maximum margin problem

$$\tilde{L}(\mathbf{a}) = \sum_{n=1}^{N} a_n - \frac{1}{2} \sum_{n=1}^{N} \sum_{m=1}^{M} a_n a_m t_n t_m k(\mathbf{x}_n, \mathbf{x}_m) \tag{10}$$

which we maximise with respect to $\mathbf{a}$, subjected to constraints

$$a_n \geq 0, \quad \text{for } n = 1, \ldots, N \tag{11}$$

$$\sum_{n=1}^{N} a_n t_n = 0 \tag{12}$$

This again takes the form of a quadratic programming problem in which we optimise a quadratic function of $\mathbf{a}$ subjected to a set of inequality constraints

Maximum margin classifiers (cont.)

Solving a quadratic program in M variables has a general complexity $\mathcal{O}(M^3)$

Going to the dual representation, we turned the original optimisation problem

- minimisation of $\arg\min_{\mathbf{w},b} \frac{1}{2}||\mathbf{w}||^2$ over M variables

into the dual problem $\tilde{L}(\mathbf{a}) = \sum_{n=1}^{N} a_n - \frac{1}{2}\sum_{n=1}^{N}\sum_{m=1}^{M} a_n a_m t_n t_m k(\mathbf{x}_n, \mathbf{x}_m)$

- N variables

For a fixed set of basis functions whose number M is smaller than the number N of data points, the move to the dual problem appears disadvantageous

However, it allows the model to be reformulated using kernels, and so the maximum margin classifier can be applied efficiently to feature spaces whose dimensionality exceeds the number of points, including infinite feature spaces

Maximum margin classifiers (cont.)

In order to classify new data points using the trained model, it suffices to evaluate the sign of $y(\mathbf{x}) = \mathbf{w}^T \phi(\mathbf{x}) + b$

This can be expressed in terms of the parameters $\{a_n\}$ and the kernel function by substituting for $\mathbf{w}$ using $\mathbf{w} = \sum_{n=1}^{N} a_n t_n \phi(\mathbf{x}_n)$ to give

$$y(\mathbf{x}) = \sum_{n=1}^{N} a_n t_n k(\mathbf{x}, \mathbf{x}_n) + b \tag{13}$$

Maximum margin classifiers (cont.)

A constrained optimisation like this satisfies **Karush-Kuhn-Tucker** conditions

$$a_n \geq 0 \tag{14}$$

$$t_n y(\mathbf{x}_n) - 1 \geq 0 \tag{15}$$

$$a_n \big(t_n y(\mathbf{x}_n) - 1\big) = 0 \tag{16}$$

Maximum margin classifiers (cont.)

For every point, either $a_n = 0$ or $t_n y(\mathbf{x}_n) = 1$

Any data point for which $a_n = 0$ will not appear in the sum in

$$y(\mathbf{x}) = \sum_{n=1}^{N} a_n t_n k(\mathbf{x}, \mathbf{x}_n) + b$$

and hence plays no role in making predictions for new points

The remaining points, called **support vectors**, satisfy $t_n y(\mathbf{x}_n) = 1$, and thus they are points that lie on the maximum margin hyperplanes in feature space

This property is central to the practical applicability of support vector machines

- Once the model is trained, a significant proportion of the data points can be discarded and only the support vectors retained

Maximum margin classifiers (cont.)

Having solved the quadratic programming problem for $\mathbf{a}$, we can determine the value of the threshold parameter b

- Note that any support vector $\mathbf{x}_n$ satisfies $t_n y(\mathbf{x}_n) = 1$

Using $y(\mathbf{x}) = \sum_{n=1}^{N} a_n t_n k(\mathbf{x}, \mathbf{x}_n) + b$ this gives

$$t_n \Big(\sum_{m \in \mathcal{S}} a_m t_m k(\mathbf{x}_n, \mathbf{x}_m) + b \Big) = 1 \tag{17}$$

$\mathcal{S}$ is the set of indices of the support vectors

We could solve this equation for b using an arbitrarily chosen support vector $\mathbf{x}_n$

Maximum margin classifiers (cont.)

A numerically more stable solution is obtained by first multiplying through by t_n, making use of $t_n^2 = 1$, and then averaging these equations over all SVs

Then solving for b gives

$$b = \frac{1}{N_{\mathcal{S}}} \sum_{n \in \mathcal{S}} \left(t_n - \sum_{m \in \mathcal{S}} a_m t_m k(\mathbf{x}_n, \mathbf{x}_m) \right) \tag{18}$$

where $N_{\mathcal{S}}$ is the total number of support vectors

Maximum margin classifiers (cont.)

The maximum margin classifier can also be expressed in terms of the minimisation of an error function, with a quadratic regulariser, of form

$$\sum_{n=1}^{N} \left(E_\infty \left(y(\mathbf{x}_n) t_n - 1 \right) \right) + \lambda ||\mathbf{w}||^2 \tag{19}$$

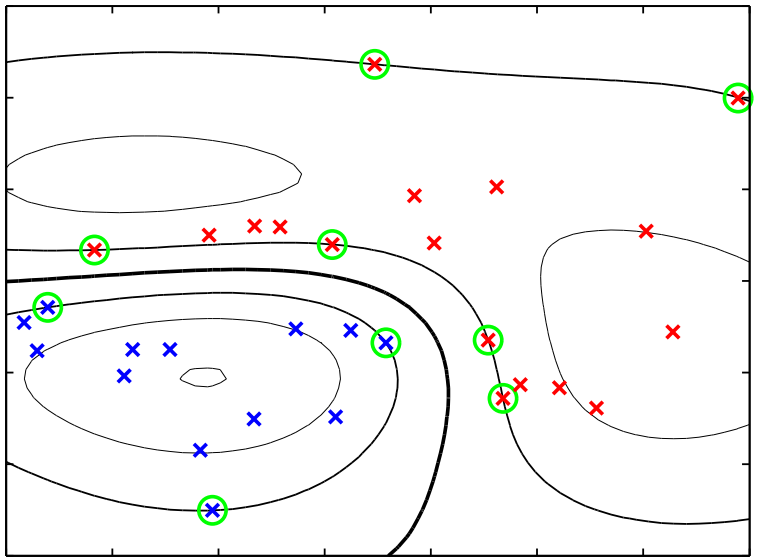
where $E_\infty(z)$ is a function that is zero if $z \geq 0$ and ∞ otherwise

- Function E_∞ also ensures that constraints $t_n(\mathbf{w}^T \phi(\mathbf{x}_n) + b) \geq 1$, with $n = 1, \ldots, N$ are satisfied
- Note that as long as the regularisation parameter satisfies $\lambda > 0$, its precise value plays no role

Maximum margin classifiers (cont.)

An example of classification resulting from training a SVM on synthetic data using a Gaussian kernel of the form $k(\mathbf{x}, \mathbf{x}') = \exp\left(-\frac{||\mathbf{x} - \mathbf{x}'||^2}{2\sigma^2}\right)$

The data are not linearly separable in the 2-D data space $\mathbf{x}$, they are linearly separable in feature space defined implicitly by the nonlinear kernel function



The training data points are perfectly separated in the original data space

- Contours of constant $y(\mathbf{x})$
- The location of the support vectors (circled) defines the position of the maximum margin hyperplane

Overlapping class distributions
Maximum margin classifiers

Overlapping class distributions

We assumed that training data are linearly separable in the feature space $\phi(\mathbf{x})$

The resulting SVM will give exact separation of the training data in the original input space $\mathbf{x}$, although the corresponding decision boundary might be nonlinear

In practice, however, the class-conditional distributions may overlap, in which case exact separation of the training data can lead to poor generalisation

We need to modify the SVM to allow some training points to be misclassified

Overlapping class distributions (cont.)

We began with a two-class classification problem using linear models of form

$$y(\mathbf{x}) = \mathbf{w}^T \phi(\mathbf{x}) + b \tag{20}$$

$\phi(\mathbf{x})$ is a fixed feature-space transformation and the bias parameter b is explicit

- The training data set comprises N input vectors $\mathbf{x}_1, , \ldots, \mathbf{x}_N$
- Target values $t_1, , \ldots, t_N$, where $t_n \in \{-1, 1\}$
- New data points $\mathbf{x}$ are classified based on the sign of $y(\mathbf{x})$

Now we do not assume anylonger that training data are linearly separable

- There may not exist a choice of parameters $\mathbf{w}$ and b such that
- $y(\mathbf{x}_n) > 0$ for points having $t_n = +1$
- $y(\mathbf{x}_n) < 0$ for points having $t_n = -1$

It is not necessarily true that $t_n y(\mathbf{x}_n) > 0$ for all N training points

Overlapping class distributions (cont.)

In the case of separable classes, we implicitly used an error function that gave infinite error if a point was misclassified and zero error if classified correctly

$$\sum_{n=1}^{N} E_\infty \Big(\underbrace{y(\mathbf{x}_n) t_n - 1}_{z} \Big) + \lambda ||\mathbf{w}||^2$$

- with $E_\infty(z)$ a function that is zero if argument $z \geq 0$ and ∞ otherwise

Then, we optimised the model parameters to maximise the margin

We modify this so that points are allowed to be on the *wrong side* of margin boundary, but with a penalty that increases with distance from the boundary

- For simplicity of the optimisation problem, it is convenient to make this penalty a linear function of this distance

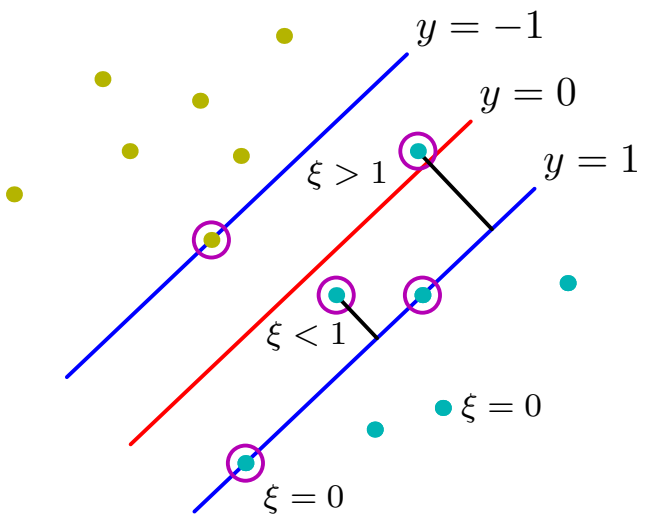
Overlapping class distributions (cont.)

We introduce **slack variables** $\xi_n \geq 0$ where $n = 1, \ldots, N$

- one slack variable for each training point

The slack variables are defined such that

- $\xi_n = 0$ for points on or inside the correct margin
- $\xi_n = |t_n - y(\mathbf{x}_n)|$ for other points



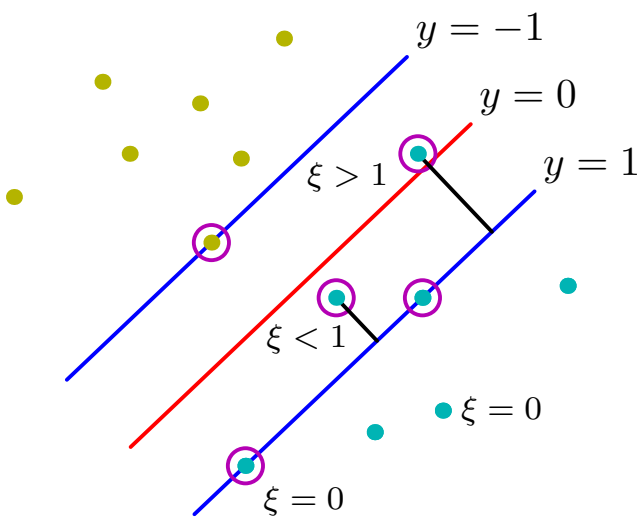
- A point on the decision boundary $y(\mathbf{x}_n) = 0$ will have $\xi_n = 1$
- Points with $\xi_n > 1$ will be misclassified

Overlapping class distributions (cont.)

Exact constraints $t_n(\mathbf{w}^T\phi(\mathbf{x}_n) + b) \geq 1$, with $n = 1, \ldots, N$ are replaced by

$$t_n(\mathbf{w}^T\phi(\mathbf{x}_n) + b) \geq 1 - \xi_n, \quad \text{with } n = 1, \ldots, N \tag{21}$$

in which the slack variables are constrained to satisfy $\xi_n \geq 0$



- Points for which $\xi_n = 0$ are correctly classified and are either on the margin or on the correct side of the margin
- Points for which $0 < \xi_n \leq 1$ lie inside the margin, but on the correct side of the decision boundary
- Points for which $\xi_n > 1$ lie on the wrong side of the decision boundary and are misclassified

Relaxing the hard margin constraint to give a **soft margin** allows some of the training set data points to be misclassified

Overlapping class distributions (cont.)

Our goal is now to maximise the margin while softly penalising points that lie on the wrong side of the margin boundary

We therefore minimise

$$\frac{1}{2}||\mathbf{w}||^2 + C\sum_{n=1}^{N}\xi_n \tag{22}$$

Parameter $C > 0$ trades off between slack variable penalty and margin

Because any point that is misclassified has $\xi_n > 1$, it follows that $\sum_n \xi_n$ is an upper bound on the number of misclassified points

The parameter C is analogous to (the inverse of) a regularisation coefficient

- Trades off between minimising errors and controlling model complexity
- In the limit $C \to \infty$, we recover the original SVM for separable data

Overlapping class distributions (cont.)

We minimise $\frac{1}{2}||\mathbf{w}||^2 + C\sum_{n=1}^{N}\xi_n$ st $t_n y(\mathbf{x}_n) \geq 1 - \xi_n$ and $\xi_n \geq 0$, $n = 1, \ldots, N$

The corresponding Lagrangian function is given

$$L(\mathbf{w}, b, \boldsymbol{\xi}, \mathbf{a}, \boldsymbol{\mu}) = \frac{1}{2}||\mathbf{w}||^2 + C\sum_{n=1}^{N}\xi_n - \sum_{n=1}^{N}a_n\Big(t_n y(\mathbf{x}_n) - 1 + \xi_n\Big) - \sum_{n=1}^{N}\mu_n\xi_n \quad (23)$$

$\{a_n \geq 0\}$ and $\{\mu_n \geq 0\}$ are Lagrange multipliers

Overlapping class distributions (cont.)

The corresponding set of KKT conditions are given for $n = 1, \ldots, N$ by

$$a_n \geq 0 \tag{24}$$
$$t_n y(\mathbf{x}_n) - 1 + \xi_n \geq 0 \tag{25}$$
$$a_n(t_n y(\mathbf{x}_n) - 1\xi_n) = 0 \tag{26}$$
$$\mu_n \geq 0 \tag{27}$$
$$\xi_n \geq 0 \tag{28}$$
$$\mu_n \xi_n = 0 \tag{29}$$

Overlapping class distributions (cont.)

We must now optimise out $\mathbf{w}$, b and $\{\xi_n\}$ using $y(\mathbf{x}) = \mathbf{w}^T\phi(\mathbf{x}) + b$

$$\frac{\partial L}{\partial \mathbf{w}} = 0 \quad \Rightarrow \quad \mathbf{w} = \sum_{n=1}^{N} a_n t_n \phi(\mathbf{x}_n) \tag{30}$$

$$\frac{\partial L}{\partial b} = 0 \quad \Rightarrow \quad \sum_{n=1}^{N} a_n t_n = 0 \tag{31}$$

$$\frac{\partial L}{\partial \xi_n} = 0 \quad \Rightarrow \quad a_n = C - \mu_n \tag{32}$$

Eliminating $\mathbf{w}$, b and $\{\xi_n\}$ from the Lagrangian, the dual Lagrangian

$$\tilde{L}(\mathbf{a}) = \sum_{n=1}^{N} a_n - \frac{1}{2}\sum_{n=1}^{N}\sum_{m=1}^{N} a_n a_m t_n t_m k(\mathbf{x}_n, \mathbf{x}_m) \tag{33}$$

Identical to what obtained in the separable case, different constraints

Overlapping class distributions (cont.)

To see what these constraints are, note that $a_n \geq 0$ (Lagrange multipliers)

Furthermore, $\frac{\partial L}{\partial \xi_n} = 0 \Rightarrow a_n = C - \mu_n$ together with $\mu_n \geq 0$ implies $a_n \leq C$

Overlapping class distributions (cont.)

We have to maximise $\tilde{L}(\mathbf{a}) = \sum_{n=1}^{N} a_n - \frac{1}{2}\sum_{n=1}^{N}\sum_{m=1}^{N} a_n a_m t_n t_m k(\mathbf{x}_n, \mathbf{x}_m)$ with respect to the dual variables $\{a_n\}$ for $n = 1, \ldots, N$, subjected to

$$a_n \in [0, C] \tag{34}$$

$$\sum_{n=1}^{N} a_n t_n = 0 \tag{35}$$

These are **box constraints** and this is an instance of a quadratic program

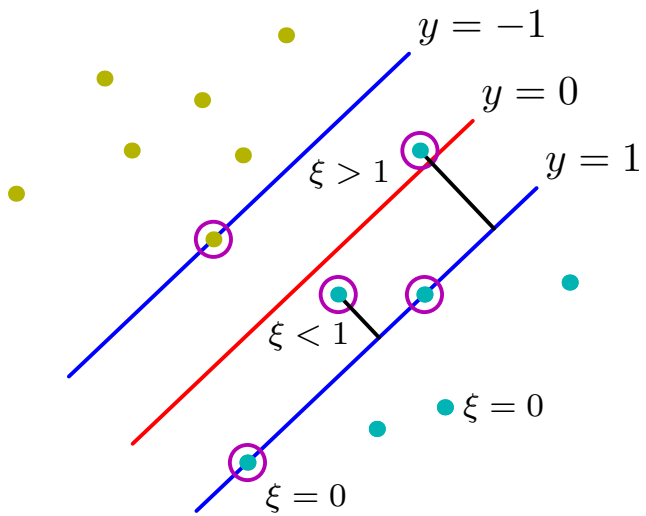
Overlapping class distributions (cont.)

We now substitute $\mathbf{w} = \sum_{n=1}^{N} a_n t_n \phi(\mathbf{x}_n)$ into $y(\mathbf{x}) = \mathbf{w}^T \phi(\mathbf{x}) + b$ to get that

- The prediction for new points is still made using

$$y(\mathbf{x}) = \sum_{n=1}^{N} a_n t_n k(\mathbf{x}, \mathbf{x}_n) + b$$

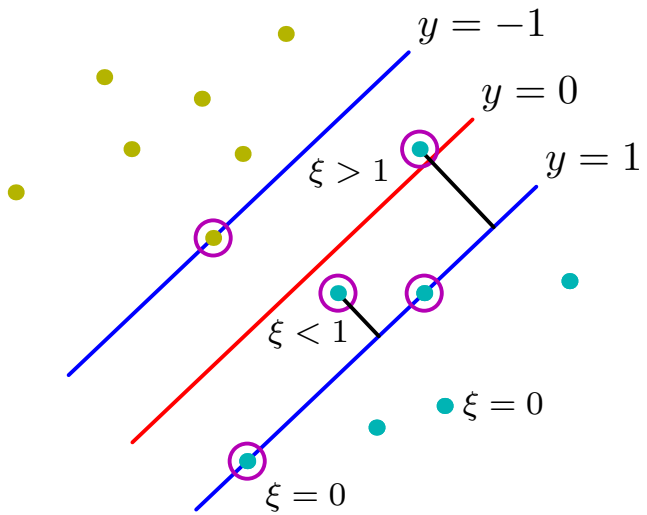
As before, a subset of points might have $a_n = 0$, and would not contribute



- The remaining points constitute support vectors
- They have $a_n > 0$ and satisfy $t_n y(\mathbf{x}_n) = 1 - \xi_n$

Overlapping class distributions (cont.)

For points with $a_n < C$, $a_n = C - \mu_n$ implies that $\mu_n > 0$, which from $\mu_n \xi_n = 0$ requires $\xi_n = 0$ and hence such points lie on the margin



Points with $a_n = C$ can lie inside the margin and can either be correctly classified if $\xi_n \leq 1$ or misclassified if $\xi_n > 1$

Overlapping class distributions (cont.)

To determine the parameter b in $y(\mathbf{x}) = \mathbf{w}^T\phi(\mathbf{x}) + b$, we note that those support vectors for which $0 < a_n < C$ have $\xi_n = 0$ so that $t_n y(\mathbf{x}_n) = 1$

Such support vectors satisfy

$$t_n\left(\sum_{m\in\mathbf{S}} a_m t_m k(\mathbf{x}_n, \mathbf{x}_m) + b\right) = 1 \tag{36}$$

A numerically stable solution can be again obtained by averaging

$$b = \frac{1}{N_{\mathcal{M}}}\sum_{n\in\mathcal{M}}\left(t_n - \sum_{m\in\mathcal{S}} a_m t_m k(\mathbf{x}_n, \mathbf{x}_m)\right) \tag{37}$$

where $\mathcal{M}$ is the set of indexes of data points having $0 < a_n < C$

Overlapping class distributions (cont.)

Predictions for new inputs are made using only the support vectors, but the training phase (i.e., determination of parameters $\mathbf{a}$ and b) uses all the data

- Need for efficient algorithms for solving the QP problem
- The use of standard algorithms is very often unfeasible

Techniques that break down the problem into a series of smaller QP problems

- **Sequential minimal optimisation** or SMO

Overlapping class distributions (cont.)

An alternative, equivalent formulation of the SVM, is known as the ν-SVM

$$\tilde{L}(\mathbf{a}) = -\frac{1}{2}\sum_{n=1}^{N}\sum_{m=1}^{N} a_n a_m t_n t_m k(\mathbf{x}_n \mathbf{x}_m) \tag{38}$$

subject to

$$a_n \in [0, \frac{1}{N}] \tag{39}$$

$$\sum_{n=1}^{N} a_n t_n = 0 \tag{40}$$

$$\sum_{n=1}^{N} a_n \geq \nu \tag{41}$$

where parameter ν replaces parameter C

Overlapping class distributions (cont.)