7.1.1 Overlapping class distributions

Although the decision boundary would be nonlinear, the resulting support vector machine will give exact separation of the training dat in the original input space x since we assumed that the training data points are linearly separable in the feature space $\phi(x)$.

In practice, however, the class-conditional distributions may overlap.

In this case, exact separation of the training data can lead to poor generalization.

So, we need to modify the support vector machine to allow some points to be misclassified ensue 일반화.

From $\sum_{n=1}^{N} E_{\infty}(y(x_n)t_n - 1) + \lambda||w||^2$, we see that in the case of separable classes,

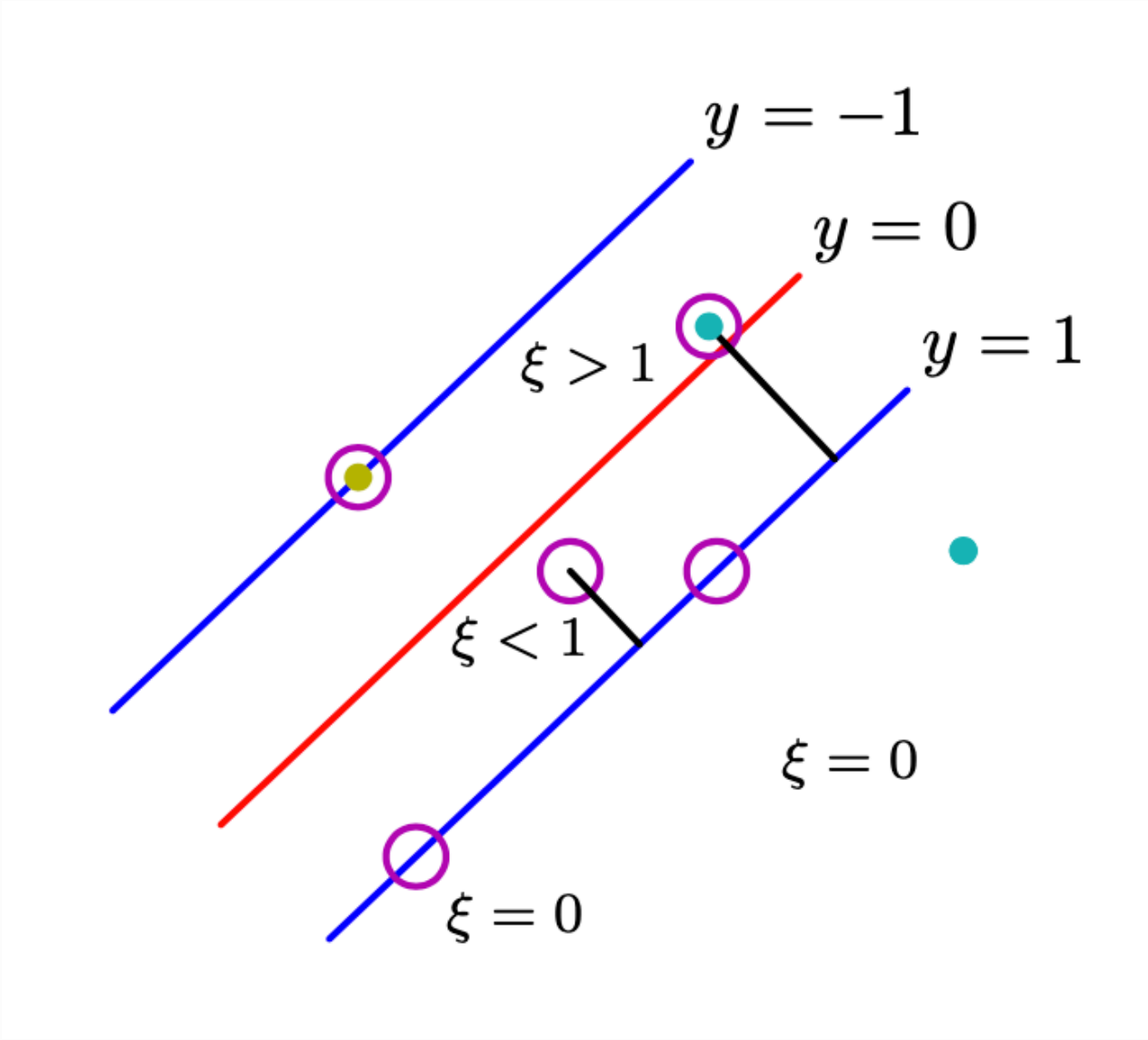
Error function gives

If a data point was misclassified, then ∞

If it was classified correctly, then 0

And then optimized the model parameters to maximize the margin.

We now modify this approach so that data points are allowed to be on the 'wrong side' of the margin boundary, but with a penalty that increases with the distance from that boundary. (적당히 틀리게 하자) and also convenient.



To do this, we introduce slack variables, $\xi_n \geq 0$ where $n = 1, .., N$, one for each training data point.

$\xi_n = 0$ on or inside the correct margin boundary and $\xi_n = |t_n - y(x_n)|$ for other points.

Data points on the decision boundary (i.e, $y(x_n) = 0$) $\rightarrow \xi_n = 1$,

Misclassified points $\rightarrow \xi_n > 1$.

$t_n(w^\top \phi(x_n) + b) \geq 1 \rightarrow t_n y(x_n) \geq 1 - \xi_n, \quad n = 1, \ldots, N \quad (7.20)$

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

Data with $\xi_n = 0$ are correctly classified (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, (틀린건 아니고) margin 사이에 박혀있는거

Those data points for which $\xi_n > 1$ lie on the wrong side of the decision boundary and are misclassified(틀린거)

(7.20) is a soft margin and allows some of the training set data points to be misclassified.

While slack variables allow for overlapping class distributions

This framework is still sensitive to outliers (because the penalty for misclassification increases ξ).

Goal : Argmax margin with soft penalty on the wrong prediction

We therefore minimize $C\sum_{n=1}^{N}\xi_n + \frac{1}{2}||w||^2 \quad (7.21)$, 앞에가 slack term, 뒤에가 margin term.

Where the parameter $C > 0$ controls the balance between the slack variable penalty and the margin.

$\sum_n \xi_n$: an upper bound on the number of misclassified points since misclassified has $\xi_n > 1$.

C : regularization coefficient.

When $C \to \infty$, we obtain support vector machine for separable data.

$Goal : \min C\sum_{n=1}^{N}\xi_n + \frac{1}{2}||w||^2$ subject to $t_n y(x_n) \geq 1 - \xi_n$ and $\xi_n \geq 0$.

The corresponding Lagrangian is given by

$$L(w,b,a) = \frac{1}{2}||w||^2 + c\sum_{n=1}^{N}\xi_n - \sum_{n=1}^{N}a_n\{t_n y(x_n) - 1 + \xi_n\} - \sum_{n=1}^{N}b_n\xi_n \qquad (7.22)$$

Where $a_n \geq 0$ and $b_n \geq 0$ are Lagrange multipliers.

The corresponding set of KKT conditions are given by

$$\begin{gathered} a_n \geq 0 \\ t_n y(x_n) - 1 + \xi_n \geq 0 \\ a_n(t_n y(x_n) - 1 + \xi_n) = 0 \\ b_n \geq 0 \\ \xi_n \geq 0 \\ b_n\xi_n = 0 \end{gathered}$$

We now optimize out $w, b, and\ \xi_n$ making use of the definition $y(x) = w\top\phi(x) + b$ (없애고 싶은 거에 대해서 편미분)

$$\frac{\partial L}{\partial w} = 0 \Rightarrow w = \sum_{n=1}^{N} a_n t_n \phi(x_n)$$

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

$$\frac{\partial L}{\partial \xi_n} \Rightarrow a_n = C - \mu_n$$

Using these results to eliminate $w, b, and\ \xi_n$ from the Lagrangian,

Dual form : $\tilde{L}(a) = \sum_{n=1}^{N} a_n - \frac{1}{2} \sum_{n=1}^{N} \sum_{m=1}^{N} a_n a_m t_n t_m k(x_m, x_m) \quad (7.32)$

Which is identical to the separable case, except that the constraints are somewhat different.

Constraint : $a_n \geq 0$ is required because these are Lagrange multipliers.

Furthermore, $a_n = C - b_n$ withe $b_n \geq 0$ implies $a_n \leq C$.

We therefore have to minimize dual form with respect to the dual variables a_n subject to $0 \leq a_n \leq C$, $\sum_{n=1}^{N} a_n t_n = 0$, where $a_n = C - b_n$ (box constraints.)

This again represents a quadratic programming problem.

If we substitute $w = \sum_{n=1}^{N} a_n t_n \phi(x_n)$ into $y(x) = w^\top \phi(x) + b$,

We see that predictions for new data points are again made by using $y(x) = \sum_{n=1}^{N} a_n t_n k(x, x_n) + b$.

As before, a subset of the data points may have $a_n = 0$, $\rightarrow$ no role int predictive model $y(x)$

Again, the remaining data points are" the support vectors".

These have $a_n > 0$ and hence from $a_n(t_n y(x_n) - 1 + \xi_n) = 0$ must satisfy $t_n y(x_n) = 1 - \xi_n$.

If $a_n < C$, then $a_n = C - b_n$,so, $b_n > 0$, $\rightarrow b_n \xi_n = 0$ means $\xi_n = 0$. So, such points lie on the margin.

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

To determine : b in $y(x) = w^\top \phi(x) + b$,

Support vectors for which $0 < a_n C$ have $\xi_n = 0$ so that $t_n y(x_n) = 1$ and

hence will satisfy $t_n(\sum_{m \in S} a_m t_m k(x_n, x_m) + b) = 1$.

Again, a numerically stable solution is obtained by averaging to give $b = \frac{1}{N_\mathrm{M}} \sum_{n \in \mathrm{M}} (t_n - \sum_{m \in S} a_m t_m k(x_n, x_m))$

where $\mathbb{M}$ denotes the set of indices of data points having $0 < a_n < C$.

An alternative, equivalent formulation of the support vector machine, known as the ν—SVM, has been proposed by 스콜코프.

This involves maximizing $\tilde{L}(a) = -\frac{1}{2}\sum_{n=1}^{N}\sum_{m=1}^{N} a_n a_m t_n t_m k(x_n, x_m)$ subject to the constraints

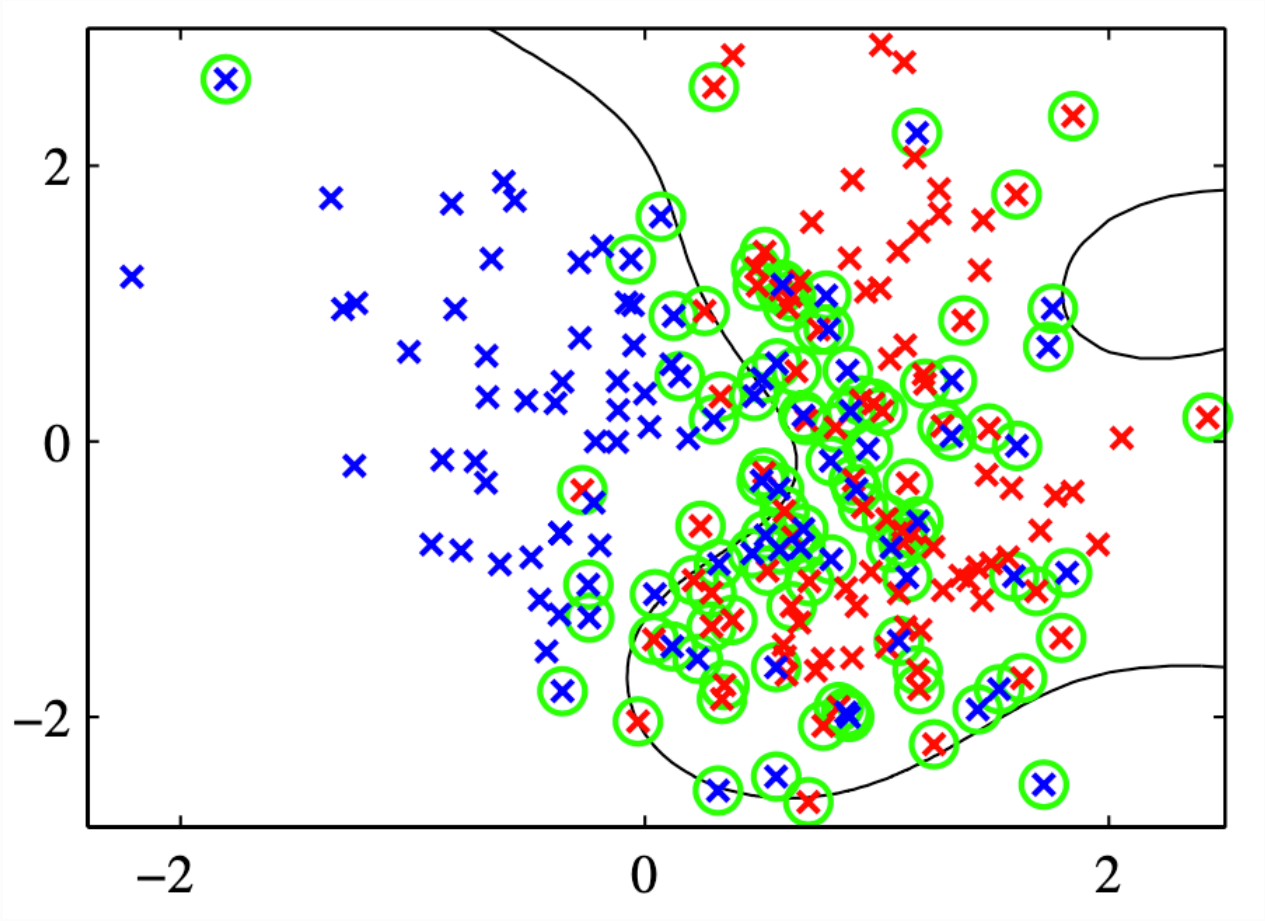
$$0 \leq a_n \leq \frac{1}{N}$$
$$\sum_{n=1}^{N} a_n t_n = 0$$
$$\sum_{n=1}^{N} a_n \geq \nu$$

This approach has the advantage that the parameter ν,which replaces C, can be interpreted as both an upper bound on the fraction of margin errors(points for which $\xi_n > 0$ and hence which lie on the wrong side of the margin boundary and which may not be misclassified) and a lower bound on the fraction of support vectors.

이 밑에는 example.

An example of the ν—SVM applied to a synthetic data set is shown in Figure.

Here Gaussian kernels of the form $\exp(-\gamma||x - x'||^2)$ have been used, with $\gamma = 0.45$.



Although predictions for new inputs are made using only the support vectors, the training phase(i.e, the determination of the parameters a and b) makes use of the whole data set, and so it is important to have efficient algorithms for solving the quadratic programming problem.

We first note that the objective function $\tilde{L}(a)$ given by (7.10) or (7.32) is quadratic and so any local optimum will also be a global optimum provided the constrains define a convex region(which they do as a consequence of being linear).

Direct solution of the quadratic programming problem using traditional techniques is often infeasible due to the demanding computation and memory requirements, and so more practical approaches need to be found.

The technique of 천킹 exploits the fact that the value of the Lagrangian is unchanged if we remove the rows and columns of the kernel matrix corresponding to Lagrange multipliers that have value zero.

This allows the full quadratic programming problem to be broken down into a series of smaller ones, whose goal is eventually to identify all of the nonzero Lagrange multipliers and discard the others.

천킹 can be implemented using protected conjugate gradients.

Although 천킹reduces the size of the matrix in the quadratic function from the number of data points squared to approximately the number of nonzero Lagrange multipliers squared, even this may be too big to fit in memory for large-scale applications.

Decomposition methods also solve a series of smaller quadratic programming problems but are designed so that each of these is of a fixed size, and so the technique can be applied to arbitrarily large data sets.

However, it still involves numerical solution of quadratic programming subproblems and these can be problematic and expensive.

One of the most popular approaches to training support vector machines is called sequential minimal optimization, or SMO.

It takes the concept of 천킹 to the extreme limit and considers just two Lagrange multipliers at a time.

In this case, the subproblem can be solved analytically, thereby avoiding numerical quadratic programming altogether.

Heuristics are given for choosing the pair of Lagrange multipliers to be considered at each step.

In practice, SMO is found to have a scaling with the number of data points that is somewhere between linear and quadratic depending on the particular application.

We have seen that kernel functions correspond to inner products in feature spaces that can have high, or even infinite, dimensionality.

By working directly in terms of the kernel function, without introducing the feature space explicitly, it might therefore seem that support vector machines somehow manage to avoid the curse of dimensionality.

This is not the case, however, because there are constraints amongst the feature values that restrict the effective dimensionality of feature space.

To see this consider a simple second-order polynomial kernel that we can expand in terms of its components

$$
\begin{aligned}
k(x, z) &= (1 + x^\top z)^2 \\
&= (1 + x_1 z_1 + x_2 z_2)^2 \\
&= 1 + 2x_1 z_1 + 2x_2 z_2 + x_1^2 z_1^2 + 2x_1 z_1 x_2 z_2 + x_2^2 z_2^2 \\
&= (1, \sqrt{2}x_1, \sqrt{2}x_2, x_1^2 \sqrt{2}x_1 x_2, x_2^2)(1, \sqrt{2}x_1, \sqrt{2}x_2, x_1^2 \sqrt{2}x_1 x_2, x_2^2)^\top \\
&= \phi(x)^\top \phi(z).
\end{aligned}
$$

This kernel function therefore represents an inner product in a feature space having six dimensions, in which the mapping from input space to feature space is described by the vector function $\phi(x)$.

However, the coefficients weighting these different features are constrained to have specific forms.

Thus any set of points in the original two-dimensional space X would be constrained to lie exactly on a two dimensional nonlinear manifold embedded in the six-dimensional feature space.

We have already highlighted the fact that the support vector machine does not provide probabilistic outputs but instead makes classification decisions for new input vectors.

베로푸로스 discuss modification

to the SVM to allow the trade-off between rkWk positive and false negative errors to be controlled.

However, if we wish to use the SVM as a module in a larger probabilistic system, then probabilistic predictions of the class label t for new inputs x are required.

To solve the issue, 플렛 has proposed fitting a logistic sigmoid to the outputs of a previously trained SVM.

Specifically, the required conditional probability is assumed to be the form

$p(t = 1|x) = \sigma(Ay(x) + B)$, where $y(x) = w^\top \phi(x) + b$.

Values for the parameters A and B are found by minimizing the cross-entropy error function defined by a training set consisting of Paris of values $y(x_n)\ and\ t_n$.

The data used to fit the sigmoid needs to be independent of that used to train the original SVM in order to avoid severe over-fitting.

This two stage approach is equivalent to assuming that the output $y(x)$ of the support vector machine represents the log-odds of x belonging to class $t = 1$.

Because the SVM training procedure is not specifically intended to encourage this,

the SVM would give a poor approximation to the posterior probabilities