1 Single-kernel Support Vector Machines

Support vector machines are based on the idea of a *fat margin*; consider a set of points divided into 2 classes which are linearly separable, a support vector machine will figure out the hyperplane that separates the two classes with the fattest margin; the fatter, the better. We may formalize the idea as follows

$$\begin{aligned} \text{minimise} \quad & \frac{1}{2}||\boldsymbol{w}||^2 \\ \text{with respect to} \quad & \boldsymbol{w}, b \\ \text{subject to} \quad & y_i(\boldsymbol{w} \cdot \boldsymbol{x}_i + b) \geq 1 \quad \forall i = 1, \ldots, m \end{aligned}$$

where m is the number of observations, $\boldsymbol{x}_i$ are the observations and y_i are the target value for each i. Graphically, this is what it means:

[insert picture]

This problem has an associated dual problem, which are as follows:

$$\begin{aligned} \text{maximise} \quad & \sum_{i=1}^{m} \alpha_i - \frac{1}{2} \sum_{i=1}^{m} \sum_{j=1}^{m} \alpha_i \alpha_j \boldsymbol{x}_i \cdot \boldsymbol{x}_j \\ \text{with respect to} \quad & \boldsymbol{\alpha} \\ \text{subject to} \quad & \boldsymbol{\alpha} \succeq \mathbf{0} \\ & \boldsymbol{\alpha}^T \mathbf{y} = 0 \end{aligned}$$

The strength of support vector machine lies on the fact that we can map each point $\boldsymbol{x}_i$ to a point on another space with different (and possibly larger) dimensionality; this space is called the feature space. We can transform dot products on the $\boldsymbol{x}_i$'s to the following

$$\boldsymbol{x}_i \cdot \boldsymbol{x}_j \rightarrow \phi(\boldsymbol{x}_i) \cdot \phi(\boldsymbol{x}_j)$$

where $\phi(.)$ is the mapping function onto the feature space. As it turns out, we don't need to specify the mapping function explicitly; what we need to do is to specify the so-called *kernel function*:

$$K(\boldsymbol{x}_i, \boldsymbol{x}_j) = \phi(\boldsymbol{x}_i) \cdot \phi(\boldsymbol{x}_j)$$

Thus we can reformulate those problems that we'd had entirely, with a slight modification. The following is the primal problem

$$\begin{aligned} \text{minimise} \quad & \frac{1}{2}||\boldsymbol{w}||^2 \\ \text{with respect to} \quad & \boldsymbol{w}, b \\ \text{subject to} \quad & y_i(\boldsymbol{w} \cdot \phi(\boldsymbol{x}_i) + b) \geq 1 \quad \forall i = 1, \ldots, m \end{aligned}$$

and this is the dual problem

$$\begin{aligned} \text{maximise} \quad & \sum_{i=1}^{m} \alpha_i - \frac{1}{2} \sum_{i=1}^{m} \sum_{j=1}^{m} \alpha_i \alpha_j K(\boldsymbol{x}_i, \boldsymbol{x}_j) \\ \text{with respect to} \quad & \boldsymbol{\alpha} \\ \text{subject to} \quad & \boldsymbol{\alpha} \succeq \mathbf{0} \\ & \boldsymbol{\alpha}^T \mathbf{y} = 0 \end{aligned}$$

There are also several kernels that may be used:

- Linear kernel: $K(\boldsymbol{x}_i, \boldsymbol{x}_j) = \boldsymbol{x}_i \cdot \boldsymbol{x}_j$
- Polynomial kernel: $K(\boldsymbol{x}_i, \boldsymbol{x}_j) = (1 + \boldsymbol{x}_i \cdot \boldsymbol{x}_j)^d$
- Gaussian kernel: $K(\boldsymbol{x}_i, \boldsymbol{x}_j) = \exp\left(-\frac{||\boldsymbol{x}_i - \boldsymbol{x}_j||^2}{2\sigma^2}\right)$
- Sigmoid kernel: $K(\boldsymbol{x}_i, \boldsymbol{x}_j) = \tanh(\beta \boldsymbol{x}_i \cdot \boldsymbol{x}_j + b)$

Not all functions can be kernels; it must satisfy Mercer's condition. Several kernels that don't satisfy Mercer's condition can in fact generalize well; the sigmoid kernel is one example.

Solving this dual problem will gives us the solution of the primal problem as well, due to the strong duality nature of the problem. Given the solution to dual problem $\boldsymbol{\alpha}^*$, which may be obtained from quadratic optimization solver, we may compute the bias

$$b = -\frac{1}{2}\left[\max_{\{i|y_i=-1\}}\left(\sum_{j=1}^{m} \alpha_j y_j K(\boldsymbol{x}_i, \boldsymbol{x}_j)\right) - \min_{\{i|y_i=+1\}}\left(\sum_{j=1}^{m} \alpha_j y_j K(\boldsymbol{x}_i, \boldsymbol{x}_j)\right)\right]$$

and the predicted class of a new data point $\boldsymbol{z}$, which is based on the sign of the following

$$\phi(\boldsymbol{z}) = b^* + \sum_{i=1}^{m} \alpha_i^* y_i K(\boldsymbol{x}_i, \boldsymbol{z}).$$

These seems computationally expensive, especially as we move on to larger datasets, but as it turns out $\boldsymbol{\alpha}^*$ is a sparse vector; there are a lot of indices i such that $\alpha_i = 0$. For any i such that $\alpha_i \neq 0$, we define the vector $\boldsymbol{x}_i$ to be a support vector. Thus the complexity of the computation reduces to the order on the number of support vectors.

Now let us consider the generalization ability of an SVM. Letting R to be the maximum radius of which a ball may contain all the datasets, and γ to be the fattest margin, we obtain the generalization inequality on SVM, which was proven by Vapnik in Statistical Learning Theory

$$\text{Generalization error} \leq \frac{R^2}{m\gamma^2}$$

thus even though the VC dimension of an SVM may be infinite (for instance, consider SVM with Gaussian kernel of small σ), we may still generalize well due to this inequality.

1.1 Soft Margin SVM

We may in fact modify SVM to allow for non-separable datasets. There are two most popular modifications, namely the ℓ_1-norm and the ℓ_2-norm regularisation.

For the ℓ_1-norm, we have the following primal problem

$$\begin{aligned} \text{minimise} \quad & \frac{1}{2}||\boldsymbol{w}||^2 + C\sum_{i=1}^{m}\xi_i \\ \text{with respect to} \quad & \boldsymbol{w}, \boldsymbol{\xi}, b \\ \text{subject to} \quad & y_i(\boldsymbol{w}\cdot\phi(\boldsymbol{x}_i)+b) \geq 1-\xi_i \quad \forall i = 1,\ldots,m \end{aligned}$$

with the dual

$$\begin{aligned} \text{maximise} \quad & \sum_{i=1}^{m}\alpha_i - \frac{1}{2}\sum_{i=1}^{m}\sum_{j=1}^{m}\alpha_i\alpha_j K(\boldsymbol{x}_i,\boldsymbol{x}_j) \\ \text{with respect to} \quad & \boldsymbol{\alpha} \\ \text{subject to} \quad & \mathbf{0} \preceq \boldsymbol{\alpha} \preceq C\mathbf{1} \\ & \boldsymbol{\alpha}^T\mathbf{y} = 0 \end{aligned}$$

again, this is solvable easily by a QP solver.

For ℓ_2-norm, we have the following primal problem

$$\begin{aligned} \text{minimise} \quad & \frac{1}{2}||\boldsymbol{w}||^2 + C\sum_{i=1}^{m}\xi_i^2 \\ \text{with respect to} \quad & \boldsymbol{w}, \boldsymbol{\xi}, b \\ \text{subject to} \quad & y_i(\boldsymbol{w}\cdot\phi(\boldsymbol{x}_i)+b) \geq 1-\xi_i \quad \forall i = 1,\ldots,m \end{aligned}$$

with the dual problem

$$\begin{aligned}
\text{maximise} \quad & \sum_{i=1}^{m} \alpha_i - \frac{1}{2} \sum_{i=1}^{m} \sum_{j=1}^{m} \alpha_i \alpha_j K(\boldsymbol{x}_i, \boldsymbol{x}_j) - \frac{1}{4C} \sum_{i=1}^{m} \alpha_i^2 \\
\text{with respect to} \quad & \boldsymbol{\alpha} \\
\text{subject to} \quad & \mathbf{0} \preceq \boldsymbol{\alpha} \preceq C\mathbf{1} \\
& \boldsymbol{\alpha}^T \mathbf{y} = 0
\end{aligned}$$

which is basically the same as the hard-margin problem, except with the modified Gram matrix of $K(\boldsymbol{x}_i, \boldsymbol{x}_i) \leftarrow K(\boldsymbol{x}_i, \boldsymbol{x}_i) + \frac{1}{2C}$.

1.2 More on Kernels

Some of the kernels that we'd had so far have parameter(s) that should be decided before the SVM gives us the result. This brings us to a question: how do we determine these parameters? Several methods exist:

- Separate the data set into training set, validation set, and test set. For each choice of parameters we have, we train the machine using the training set, and find the error on validation set; this is called the validation error. Then we pick the parameters that produces the smallest validation error.
- Use cross validation; the most famous example is the leave-one-out. In this procedure, for each data point, we extract that data point out of the training set, and train the machine with the rest of $(m-1)$ points. We evaluate the hypothesis versus the singled-out point; the expected value of the error is known as the cross-validation error. We don't need to single out one point; in fact k-fold cross validation is common, when we take k points out of the training sets. Practitioners recommend the size of k to be between $m/5$ and $m/10$.
- Minimise the generalization bound; by varying the parameters and training the machine, we may obtain the formula for R and γ. They are as follows:

$$R^2 = \max_i \left\{ \sum_{j=1}^{m} \sum_{k=1}^{m} \lambda_j \lambda_k K(\boldsymbol{x}_j, \boldsymbol{x}_k) - 2 \sum_{j=1}^{m} \lambda_j K(\boldsymbol{x}_i, \boldsymbol{x}_j) + K(\boldsymbol{x}_i, \boldsymbol{x}_i) \right\}$$

$$\frac{1}{\gamma^2} = \sum_{i=1}^{m} \alpha_i^*$$

Thus we can minimize them as the parameters vary, and choose the parameters with the smallest generalization error bound.

- Use multiple kernel learning; for instance, use a linear combination of chosen kernels, and solve the optimization problem as we allow the weights to vary. This is the purpose of our final year project.

2 Multiple Kernel Learning

Now consider a set of kernels $\mathcal{K}$. For each kernel $K \in \mathcal{K}$, we may define the following function

$$\omega(K) = \max_{\boldsymbol{\alpha}} \left\{ \sum_{i=1}^{m} \alpha_i - \frac{1}{2} \sum_{i=1}^{m} \sum_{j=1}^{m} \alpha_i \alpha_j y_i y_j K(\boldsymbol{x}_i, \boldsymbol{x}_j) : \boldsymbol{\alpha}^T \boldsymbol{y} = 0, \boldsymbol{\alpha} \succeq \mathbf{0} \right\}.$$

Now by stating the constraint explicitly and assuming that the dataset is linearly separable on the feature space, we have the following problem

$$\begin{aligned} \min_{K \in \mathcal{K}} \omega(K) = \min_{K \in \mathcal{K}} \max_{\boldsymbol{\alpha}} & \sum_{i=1}^{m} \alpha_i - \frac{1}{2} \sum_{i=1}^{m} \sum_{j=1}^{m} \alpha_i \alpha_j K(\boldsymbol{x}_i, \boldsymbol{x}_j) \\ \text{with respect to} \quad & K, \boldsymbol{\alpha} \\ \text{subject to} \quad & \mathbf{0} \preceq \boldsymbol{\alpha} \preceq C\mathbf{1} \\ & \boldsymbol{\alpha}^T \mathbf{y} = 0 \end{aligned}$$