Support Vector Machines

1 Hypothesis

Consider the domain $\mathcal{X} = \mathbb{R}^D$ and the codomain $\mathcal{Y} = \{-1, 1\}$. In Support Vector Machines, we are trying to find an optimal hypothesis $h_{\mathbf{w},w_0} : \mathcal{X} \to \mathcal{Y}$ in the hypothesis class

$$\mathcal{H} = \{h_{\mathbf{w},w_0} : h_{\mathbf{w},w_0}(\mathbf{x}) = \operatorname{sgn}(w^\top x + w_0)\}$$

which maximizes the geometric margin from a hyperplane to the support vectors in the dataset (for more information read **SVM Margins**).

2 Optimization

By requiring that the length of the functional margin be at least one and be exactly one for support vectors (since it can be of arbitrary length), we remove the degree of freedom regarding the length of $\mathbf{w}$ and get that the geometric margin will be $\frac{2}{\|\mathbf{w}\|}$. As such, we can now optimize over the direction of $\mathbf{w}$ only. This requirement also introduces the constraint $y_i(\mathbf{w}^\top \mathbf{x}_i + w_0) \geq 1$ for all data points. Since we want to maximize the geometric margin subject to these constraints, we want to solve

$$\arg\max_{\mathbf{w}} \frac{2}{\|\mathbf{w}\|} = \arg\min_{\mathbf{w}} \frac{\|\mathbf{w}\|^2}{2} = \arg\min_{\mathbf{w}} \frac{1}{2}\mathbf{w}^\top \mathbf{w}$$

$$\text{s.t.} \quad \forall(\mathbf{x}_i, y_i) \in S, \quad y_i(\mathbf{w}^\top \mathbf{x}_i + w_0) \geq 1$$

This also gives some insight as to why we choose 1 for the functional margin: since the equation is quadratic in $\mathbf{w}$, having $\frac{1}{2}$ as a denominator makes the derivative simpler.

This type of optimization is called *quadratic optimization*, since we are optimizing a quadratic function subject to linear constraints. By adding Lagrange multipliers α_i to each constraint, it can be shown that solving this optimization is equivalent to solving the formula below

$$\arg\max_{\alpha} \sum_{i=1}^{N} \alpha_i - \frac{1}{2}\sum_{i=1}^{N}\sum_{j=1}^{N} \alpha_i \alpha_j y_i y_j \mathbf{x}_i^\top \mathbf{x}_j$$

$$\text{s.t} \quad \forall \alpha_i, \quad \sum_{i=1}^{N} \alpha_i y_i = 0, \quad \alpha_i \geq 0$$

The α_i terms offer some insight as to why support vectors exist: those terms are only non-zero for support vectors, indicating that only support vectors contribute to determining the margin and the remainder of the data is in fact irrelevant.

Once α is known, $\mathbf{w}$ is given by

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

To finally find w_0, all that remains is plugging in a labeled training point to the constraint $y_i(\mathbf{w}^\top \mathbf{x}_i + w_0) \geq 1$.

3 Soft Margin

For most data, it is not possible to strictly enforce the constraint $y_i(\mathbf{w}^\top \mathbf{x}_i + w_0) \geq 1$. Thus, it is usually useful to "relax" this constraint and allow for a outliers. To do this, we introduce slack variables C and ξ_i, updating our objective function to

$$\arg\min_{\mathbf{w},\xi} \frac{1}{2}\mathbf{w}^\top \mathbf{w} + C\sum_{i=1}^{N} \xi_i$$

$$\text{s.t.} \quad \forall i, \quad y_i(\mathbf{w}^\top \mathbf{x}_i + w_0) \geq 1 - \xi_i, \quad \xi_i \geq 0$$

ξ_i represents the distance from the data point in question to its class support vector "boundary" (note that we are essentially allowing for smaller or even negative functional margins for some points). As such, if $\xi_i < 1$, the point is still within its class "region", but has a smaller functional margin; if $\xi_i = 1$, it is right at the division between the "regions"; if $\xi_i > 1$, it is in the other class's "region".

C determines the amount of regularization. Due to the modified constraints, C determines how relevant the ξ_i parameters are. If C is very small, $\sum \xi_i$ will have very little impact on the objective function, and this makes it so that larger ξ_i are allowed. If C is large, $\sum \xi_i$ will have a large impact, penalizing larger ξ_i. Thus, C is inversely proportional to the amount of regularization.

4 Kernels

Most of the time, data is not linearly separable. One effective approach to tackling this issue is to transform the original feature space into a higher-dimensional space, which may in fact have a hyperplane that separates the data.

Instead of directly computing this transformation, a characteristic of the quadratic optimization equation can be exploited to reduce computation. Imagine the mapping ϕ of the original feature space to a higher-dimensional space. The equivalent quadratic optimization formula would then be

$$\arg\max_{\alpha} \sum_{i=1}^{N} \alpha_i - \frac{1}{2}\sum_{i=1}^{N}\sum_{j=1}^{N} \alpha_i \alpha_j y_i y_j \phi(\mathbf{x}_i)^\top \phi(\mathbf{x}_j)$$

Notice that ϕ only appears in an inner product. Because of this, we could use a function which computes $\phi(\mathbf{x}_i)^\top \phi(\mathbf{x}_j) = \langle \phi(\mathbf{x}_i), \phi(\mathbf{x}_j) \rangle$ directly. This function is called a "kernel", and will therefore be defined as $k(\mathbf{x}, \mathbf{x}') = \langle \phi(\mathbf{x}), \phi(\mathbf{x}') \rangle$. The quadratic optimization formula then becomes

$$\arg\max_{\alpha} \sum_{i=1}^{N} \alpha_i - \frac{1}{2} \sum_{i=1}^{N} \sum_{j=1}^{N} \alpha_i \alpha_j y_i y_j k(\mathbf{x}_i, \mathbf{x}_j)$$

and the hypothesis becomes (see **Kernelized SVM Hypothesis** for the derivation)

$$h_{\alpha,w_0}(\mathbf{x}) = \operatorname{sgn}\left(w_0 + \sum_{i=1}^{N} \alpha^{(i)} y^{(i)} k(\mathbf{x}^{(i)}, \mathbf{x}) \right)$$

These kernel functions "implicitly" transform the feature space since they skip the actual transformation and directly compute the inner products, reducing the computation necessary. Further, since α_i is 0 for non-support vectors, the kernel function need not be computed for them, reducing computation even more.