Support Vector Machines

A support vector machine is a supervised learning algorithm to find an optimal separating hyperplane that maximises the margin between the training samples classified into groups.

The distance between the two closest opposing examples on either side of the hyperplane are called margins. The optimal hyperplane maximises the margin.

The equation of the hyperplane is given by w.x = 0. This representation allows generalisation into more dimensions, with vector w always normal to the hyperplane.

We add a bias w0, which determines the plane's intersection with the vertical axis.

The margin is equal to m = 2*||p||, where p is the distance from the point to the hyperplane. To find the biggest margin possible, we choose two hyperplanes that separate the data and maximise their distance.

The Algorithm

1. Initially data in a dataset D is coupled as <**x**,y> where **x** is a vector and y can take values of -1 and 1.
2. The data must be linearly separable. To find the hyperplanes, we take a general hyperplane **w.x** + b = 0 and two separating hyperplanes **w.x** + b = d and **w.x** + b = -d and assume d = 1.
3. After hyperplane selection, we subject them to constraints as **w.x** + b >= 1 for **x** having class 1 and **w.x** + b <= 1 for **x** having class -1.
4. Combining these two hyperplanes and multiplying by y, we get a single equation representing both these planes : y(**w.x** + b) >=1.
5. Now, we find the margin between the two planes. For this we assume a point x0 on plane **w.x** + b = -1 and x1 on **w.x** + b = 1. Now m is the perpendicular for x0 to **w.x** + b = 1. So we can represent the margin p = **w**/||**w**|| and **k** = m**p** where k is perpendicular to **w.x** + b = -1.
6. The distance is given by z0 = x0 + k.
7. Since z0 lies on H1 which is **w.x** + b = 1, **w.z0** + b = 1.

   Which means, **w.(x0 + k)** + b = 1.

   Implying, **w.x** + **m** $||\mathbf{w}||^2$ /||**w**|| + b = 1.

This gives m = 2/||w||.

8. We use the principle of duality and say we maximise ||w|| to minimise m. Solution to this constraint gives the optimal separating hyperplane.

The principle of duality says that an optimisation problem can be viewed from a primal perspective or a dual perspective, where the solution to the dual provides a lower bound to the solution of the primal.

The main problem of the SVM is to find a feasible solution to minimise m subject to y(**w.x** + b) = 1.

We use Lagrange's theorem as the min(f(x,y)) subject to g(x,y) = 0 are obtained when their gradients point in the same direction.

Or, f(x,y) = k.g(x,y) where k is the multiplier.

We can write the Lagrangian L(x,y,k) as

L(x,y,k) = f(x,y) - k.g(x,y)

We need to find the derivative of L to give the points where f and g are parallel.

So we do,

L(x,y,k) = 2/||w|| - k(y(**w.x** + b))

Taking the differential on both sides and equating the partials to 0, we get

L = ½ (sum(k.y.x)^2) - sum(k.y.x * (sum(k.y.x)) + sum(k)

The values of w and b for which L = 0 give the supporting hyperplanes.

Karush Kuhn Tucker Conditions

If these conditions are met, the SVM provides an optimal solution

Stationarity condition

$$\nabla L_w = w - \sum_{i=1}^{m} \alpha_i y_i x_i = 0$$

$$\frac{\partial L}{\partial b} = - \sum_{i=1}^{m} \alpha_i y_i = 0$$

Primal f easibility condition:

$$y_i\left(w.x_i + b\right) - 1 \geq 0$$

Dual f easibility condition:

$$\alpha_i \geq 0$$

Complementary slackness condition:

$$\alpha_i\left[y_i\left(w.x_i + b\right) - 1\right] = 0$$

SVMs work only on linearly separable data (at least vanilla SVMs). To use non linearly separable data, we use a kernel trick transformation to project the data into higher dimensions.