Training Support Vector Machines in 1D

Yang Su T. M. Murali Vladimir Pavlovic Simon Kasif

December 8, 2002

Abstract

Given n numbers belonging to two classes, this note describes an $O(n \log n)$ algorithm for training a support vector machine (SVM) on these numbers.

Let S be a set of n points in $\mathbb{R}^d$ where each point $x_i \in S$ has a label $y_i \in \{-1, 1\}$. We say that a point with label 1 is *positive* and that a point with label -1 is *negative*. We are interested in the case when there may not be any hyperplane that separates the positive points from the negative points. For each point $x_i \in S$, we introduce a slack variable ξ_i. We want a hyperplane (w, b) (a point x on the hyperplane satisfies $w \cdot x + b = 0$) such that the following inequalities hold:

$$\begin{aligned} y_i(x_i \cdot w + b) &\geq 1 - \xi_i, \qquad \forall i & (1) \\ \xi_i &\geq 0, \qquad \forall i & (2) \end{aligned}$$

A point x_i is incorrectly classified iff $\xi_i > 1$. The function we want to minimise is $\|w^2\|/2 + c\sum_i \xi_i$, where c is a suitable (user-defined) constant. Introducing a Lagrange multiplier α_i for each constraint in (1) and a Lagrange multiplier μ_i for each constraint in (2), we have the following (primal) Lagrangian to minimise:

$$L_P = \frac{\|w^2\|}{2} + c\sum_i \xi_i - \sum_i \alpha_i \left(y_i(x_i \cdot w + b) - 1 + \xi_i\right) - \sum_i \mu_i \xi_i \qquad (3)$$

A *support vector* is a point x_i that satisfies (1) (has $\xi_i = 0$) and has $\alpha_i > 0$ in the solution.[1]

The Karush-Kuhn-Tucker (KKT) conditions imply that the solution that achieves the minimum satisfies the following conditions (w_j is the jth component of w and x_{ij} is the j coordinate of x_i):

$$\begin{aligned} \frac{\partial L_p}{\partial w_j} = w_j - \sum_i \alpha_i y_i x_{ij} &= 0, \qquad \forall 1 \leq j \leq d & (4) \\ \frac{\partial L_p}{\partial b} = -\sum_i \alpha_i y_i &= 0, \qquad \forall 1 \leq i \leq n & (5) \\ \frac{\partial L_p}{\partial \xi_i} = c - \alpha_i - \mu_i &= 0, \qquad \forall 1 \leq i \leq n & (6) \\ \alpha_i, \xi_i, \mu_i &\geq 0, \qquad \forall 1 \leq i \leq n & (7) \\ \alpha_i \left(y_i(x_i \cdot w + b) - 1 + \xi_i\right) &= 0 \qquad \forall 1 \leq i \leq n & (8) \\ \mu_i \xi_i &= 0 \qquad \forall 1 \leq i \leq n & (9) \end{aligned}$$

[1] Note that points that satisfy (1) can have $\alpha_i = 0$.

1 Observations

We can make several observations based on the KKT conditions. Note that if x_i is correctly classified, then $0 \leq \xi_i \leq 1$ and $y_i(x_i \cdot w + b) \geq 0$.

Observation 1 *If x_i is correctly classified and satisifies $y_i(x_i \cdot w + b) > 1$, then $\xi_i = \alpha_i = 0$.*

Proof: By definition, $\xi_i = 0$ for a correctly-classified point x_i that satisifies $y_i(x_i \cdot w + b) \geq 1$. Equation (8) implies that $\alpha_i = 0$. □

Observation 2 *For a point x_i, if $\xi_i > 0$ then $\mu_i = 0, \alpha_i = c$, and $\xi_i = 1 - y_i(x_i \cdot w + b)$.*

Proof: If $\xi_i > 0$, then (9) and (6) imply that $\mu_i = 0$ and $\alpha_i = c$. Further, (8) implies that $y_i(x_i \cdot w + b) - 1 + \xi_i = 0$. □

Thus, $\alpha_i = c$ for all points that have $\xi_i > 0$. These points also satisfy the equation $y_i(x_i \cdot w + b) < 1$. These observations provide the values of α_i and ξ_i in the optimal solution for all points x_i except for those that satisify $y_i(x_i \cdot w + b) = 1$. Only the support vectors amongst these points have $\alpha_i > 0$. Using (5), we can now prove the following observation about the sum of the α values of the support vectors:

Observation 3 *Let α^+ be the total value of the α_i's of the positive support vectors, let α^- be the total value of the α_i's of the negative support vectors, let n^+ be the number of positive points with $\xi_i > 0$ and let n^- be the number of negative points with $\xi_i > 0$. Then,*

$$\alpha^+ - \alpha^- + c(n^+ - n^-) = 0 \tag{10}$$

We now state the key observations that apply to points in one dimension.

Observation 4 *If all the points in S are one-dimensional, all positive support vectors have the same coordinate (a similar condition holds for the negative support vectors).*

Proof: If x_i is a support vector, then by definition $y_i(x_i \cdot w + b) = 1$. In one dimension, given y_i, w, and b, there is only one value of x_i that satisfies this equation. □

We can now prove the following corollary to Observation 3:

Observation 5 *If all the points in S are one-dimensional, then $n^+ = n^-$ and $\alpha^+ = \alpha^-$.*

Proof: Suppose there is more than one positive support vector (Observation 4 implies that all these points have the same coordinate). We obtain an identical solution by setting the alpha value for one of these support vectors to α^+ and the rest to 0. Thus, we can assume that there is only one positive support vector and one negative support vector.

If $n^+ \neq n^-$, then Observation 3 implies that $|\alpha^+ - \alpha^-| \geq c$. The definition of support vectors implies that $\alpha^+, \alpha^- > 0$. Combining (6) and (7), we have $\alpha^+, \alpha^- \leq c$. Therefore, if $|\alpha^+ - \alpha^-| \geq c$, then either α^+ or α^- must be 0, which is a contradiction. □

2 Algorithm

We assume that positive points lie to the left of negative points. In this scenario, if a point p is the positive (respectively, negative) support vector, then n^+ (respectively, n^-) is the number of positive (respectively, negative) points to its right (respectively, left). These points have positive value of ξ_i in the optimal solution. If we know that p is the positive support vector, then there is exactly one point q that satisfies Observation 5. See Figure 1.

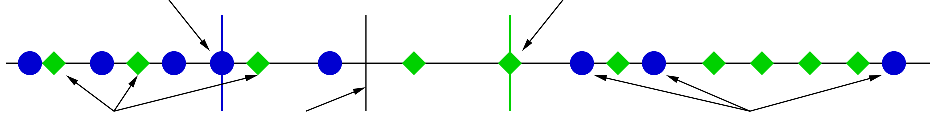

Figure 1: SVMs in one dimension. Positive points are circles and negative points are diamonds. In this figure, $n^+ = n^- = 3$.

The training algorithm in one dimension uses this observation. We first set up some notation to ease the description of the algorithm. Suppose that there are n positive points and m negative points. Let p_i be the ith positive point in sorted order from left to right. We abuse notation and use p_i to also denote the coordinate of this point. Let $d_i^+ = \sum_{i<k\leq n} p_k$ denote the sum of the coordinates of the positive points to the right of p_i. The number of such points is $n-i$. Similarly, let q_j be the jth negative point in sorted order from right to left (with a corresponding abuse of notation) and let $d_j^- = \sum_{j<k\leq m} q_k$. If p_i is the positive support vector in the optimal solution, then q_i is the negative support vector. Only the positive points to the right of p_i and the negative points to the left of q_i have values of $\alpha = c$. Using these facts and assuming that p_i and q_i are the optimal support vectors, we can calculate the values of w, α^+, α^-, and $\sum_k \xi_k$ as follows:

(a) A support vector has slack variable equal to 0. Therefore, (1) implies that $p_i \cdot w + b = 1$ and $q_i \cdot w + b = -1$, which means that $w = 2/(p_i - q_i)$.

(b) Equation (5) implies that $w = \alpha^+ p_i - \alpha^- q_i + c\sum_{k>i} p_k - c\sum_{k>i} q_k = \alpha^+(p_i - q_i) + c(d_i^+ - d_i^-)$, which implies that $\alpha^+ = \alpha^- = \left(w - c(d_i^+ - d_i^-)\right)/(p_i - q_i)$, and

(c) Observation 2 implies that $\sum_k \xi_k = \sum_{k>i} (1 - (p_k \cdot w + b)) + \sum_{k>i} (1 + (q_k \cdot w + b)) = 2(n - i) - w(d_i^+ - d_i^-)$.

Thus, given the support vectors, the rank of the support vectors in the sorted order of points, and the corresponding d^+ and d^- values, we can calculate the optimal value of the Lagrangian L_p in $O(1)$ time. We can now describe the algorithm.

1. Sort the positive points from left to right.
2. For i ranging from n down to 1, compute d_i^+ using the equation $d_i^+ = d_{i+1}^+ + p_{i+1}$.
3. Sort the negative points from right to left.
4. For j ranging from m down to 1, compute d_j^- using the equation $d_j^- = d_{j+1}^- + q_{j+1}$.
5. For i ranging from 1 to n,

(a) Set p_i to be the positive support vector.
(b) Set q_i to be the negative support vector.
(c) Compute w_i.
(d) Compute $b_i = (1 - p_i)/w_i$.[2]
(e) Compute $\sum_k \xi_k$ as indicated above.
(f) Set $L_i = w^2/2 + c\sum_k \xi_k$.

6. The optimal solution corresponds to the value i that minimises L_i.

After some pre-processing (Step 1 to Step 4), the algorithm tries every positive point as a candidate for being the positive support vector (Step 5a). For each such point, it determines the corresponding negative support vector (Step 5b), and then computes the values of w, b, and the sum of the slack variables (Step 5c to Step 5e). Finally, in Step 5f, it computes the value of the Lagrangian L_p for the current choice of support vectors. The minimum value of L_p over all the choices of the support vectors provides the final solution.

We can execute the sorting steps (Steps 1 and 3 in $O(n \log n)$ time. The time taken to calculate the d^+ and d^- values in Steps 2 and 4 is $O(n)$ (a prefix sum computation). Each iteration of the main loop (Step 5) takes $O(1)$ time. Thus, the overall algorithm runs in $O(n \log n)$ time.

[2] We can use any point whose α value is not zero to calculate b_i.