Sequential Minimal Optimization Method to Solve the Support Vector Machine Problem

Enzhi Li

July 16, 2018

I Introduction

In this article, I have given a detailed derivation of the support vector machine (SVM) algorithm. In the same article, I also tried to solve the SVM problem using interior point method. The disadvantage of the interior point method is that this method requires the storage and manipulation of a square dense matrix, the dimension of which is equal to the number of data points. In the real world application, the number of data points could be millions, and the mere storage of the matrix would cause out of memory problem, let alone the manipulation thereof. Here, in this article, I am going solve the SVM problem using a fast and efficient algorithm developed by John Platt in 1998 in this paper. This fast and efficient method, called sequential minimal optimization (SMO), reduces the original huge optimization problem to a set of small and simple problems, so simple that these mini problems can be solved exactly without resorting to any unwieldy numerical libraries. Here, I am going to describe the SMO algorithm in detail, implement this algorithm, and present the experimental results on synthetic two dimensional data.

The organization of this article is as follows. In section II, I am going to give a brief summary of the SVM algorithm. Section III focuses on the SMO algorithm, and section IV presents the program implementation of the algorithm, together with the experimental results.

II Support vector machine (SVM)

SVM is used to classify data points of the form $(\boldsymbol{x}_i, y_i), i = 1, 2, ..., N$. Here, $\boldsymbol{x}_i \in \mathbb{R}^N$ is the feature vector and $y_i \in \{-1, 1\}$ is the data label. For a linearly separable set of data, we can always find a hyperplane that satisfies the condition that $\beta^T x + \beta_0 = 0$. With this separating hyperplane, the points the lie on one side of this plane satisfy $\beta^T x + \beta_0 > 0$, whereas the points that lie on the other side satisfy $\beta^T x + \beta_0 < 0$. With some proper rescaling, we can make the positively labeled points satisfy the condition that $\beta^T x + \beta_0 \geq 1$, and the negatively labeled points satisfy $\beta^T x + \beta_0 \leq -1$. For any data set $(\boldsymbol{x}_i, y_i)$, there are more than one set of parameters β, β_0 that satisfy these conditions. However, there is one set of parameter for which the distance between the two planes $\beta^T + \beta_0 = \pm 1$ is largest. The hyperplane corresponding to this set of parameters is called the optimal separating line (plane). The training of SVM using data is just the determination of this set of optimal parameters.

The determination of the optimal parameters can be converted to this constrained optimization problem:

$$\begin{aligned} & \max_{\beta} \frac{1}{2}||\beta||^2 \\ \text{s.t.} \quad & -y_i(\beta^T x_i + \beta_0) \leq -1, i = 1, 2, ..., N \end{aligned} \tag{1}$$

This problem can be further transformed to its Lagrangian dual form, which is,

$$\begin{aligned} & \max_{\alpha} \sum_{i=1}^{N} \alpha_i - \frac{1}{2} \sum_{i=1}^{N} \sum_{j=1}^{N} \alpha_i y_i \alpha_j y_j x_i^T x_j \\ \text{s.t.} \quad & \alpha_i \geq 0, \forall i, \\ & \sum_{i=1}^{N} \alpha_i y_i = 0, \end{aligned} \tag{2}$$

This Lagrangian dual form is easier to solve than the original problem. It is a convex quadratic optimization problem with linear constraints. A fast and efficient algorithm is the sequential minimal optimization algorithm. Next, I am going to present the derivation of this algorithm.

III Sequential minimal optimization (SMO)

As can be seen from the previous section, SVM can be finally converted to a quadratic optimization problem with linear constraints. We know that the quadratic problem can be solved exactly by hand when the number of variables is not too large. SMO takes advantage of this and decomposes the original quadratic problem into a series of small and manageable mini quadratic problems. The original has N variables, $\alpha_i, i = 1, 2, ..., N$. SMO considers two of these variables each time. The reason that SMO considers only two of these $\alpha's$ is that the original problem has a linear constraint, which is $\sum_{i=1}^{N} \alpha_i y_i = 0$. As a result, we must vary at least two of these $\alpha's$ simultaneously for the new $\alpha's$ still to satisfy this constraint. In SMO, we simply choose the smallest possible number of variation variables, that is, two.

As shown in the previous section, our target function for the optimization problem is

$$g(\alpha) = \sum_{i=1}^{N} \alpha_i - \frac{1}{2} \sum_{i=1}^{N} \sum_{j=1}^{N} \alpha_i y_i \alpha_j y_j x_i^T x_j \tag{3}$$

SMO decomposes this quadratic function into two parts. One part depends on the variation variables, which we denote as α_1, α_2, and the other part is independent of the variation variables. The part that depends on the variation variables is

$$\begin{aligned} f(\alpha_1, \alpha_2) &:= g^{(v)} \\ &= \frac{1}{2}\alpha_1^2 y_1^2 x_1^T x_1 + \frac{1}{2}\alpha_2^2 y_2^2 x_2^T x_2 + \alpha_1 \alpha_2 y_1 y_2 x_1^T x_2 \\ &+ \alpha_1 y_1 x_1^T \sum_{i=3}^{N} \alpha_i y_i x_i + \alpha_2 y_2 x_2^T \sum_{i=3}^{N} \alpha_i y_i x_i - \alpha_1 - \alpha_2 \end{aligned} \tag{4}$$

The part that is independent of α_1, α_2 is

$$g^{(c)} = \frac{1}{2} \sum_{i,j=3}^{N} \alpha_i \alpha_j y_i y_j x_i^T x_j - \sum_{i=3}^{N} \alpha_i = \text{const} \tag{5}$$

Thus, the original quadratic optimization problem can be rewritten as

$$\begin{aligned} &\max_{\alpha_1, \alpha_2} f(\alpha_1, \alpha_2) + \text{const} \\ \text{s.t.} \quad &0 \leq \alpha_i \leq C, i = 1, 2; \\ &\alpha_1 y_1 + \alpha_2 y_2 = \gamma = -\sum_{i=3}^{N} \alpha_i y_i \end{aligned} \tag{6}$$

Here, I have used the soft margin SVM, and thus all the $\alpha_i, i = 1, 2, ..., N$ are restricted within the range $[0, C]$. From the equation $\alpha_1 y_1 + \alpha_2 y_2 = \gamma$, we can solve α_2 from α_1, and *vice versa*. That is,

$$\begin{aligned} \alpha_1 &= y_1 \gamma - y_1 y_2 \alpha_2; \\ \alpha_2 &= y_2 \gamma - y_1 y_2 \alpha_1 \end{aligned} \tag{7}$$

Define $s = y_1 y_2$. Note that $y_i \in \{-1, 1\}$, and thus $y_i^2 = 1, s^2 = 1$. I can represent α_1 in terms of α_2 as $\alpha_1 = y_1 \gamma - s\alpha_2$. Plug this into $f(\alpha_1, \alpha_2)$, and I get

$$\begin{aligned} \tilde{f}(\alpha_2) &:= f(\gamma y_1 - s\alpha_2, \alpha_2) \\ &= \frac{1}{2}\Big(\gamma^2 + \alpha_2^2 - 2\alpha_2 \gamma y_2\Big) x_1^T x_1 + \frac{1}{2}\alpha_2^2 x_2^T x_2 \\ &+ (\gamma y_2 \alpha_2 - \alpha_2^2) x_1^T x_2 + \Big(\gamma x_1^T + \alpha_2 y_2 (x_2^T - x_1^T)\Big) \sum_{i=3}^{N} \alpha_i y_i x_i \\ &- \gamma y_1 + (s - 1)\alpha_2 \end{aligned} \tag{8}$$

This is quadratic function, and can be rewritten into the standard form as

$$\begin{aligned}
\tilde{f}(\alpha_2) &= \frac{1}{2}(x_1^T x_1 + x_2^T x_2 - 2x_1^T x_2)\alpha_2^2 \\
&+ \Big(- \gamma y_2 x_1^T x_1 + \gamma y_2 x_1^T x_2 + y_2(x_2^T - x_1^T) \sum_{i=3}^{N} \alpha_i y_i x_i + s - 1 \Big) \alpha_2 \\
&+ \frac{1}{2}\gamma^2 x_1^T x_1 + \gamma x_1^T \sum_{i=3}^{N} \alpha_i y_i x_i - \gamma y_1
\end{aligned} \tag{9}$$

It is easy to see that the coefficient of the quadratic term is semi-positive-definite, since

$$x_1^T x_1 + x_2^T x_2 - 2x_1^T x_2 = (x_1 - x_2)^T (x_1 - x_2) \geq 0 \tag{10}$$

Therefore, except for the case where $x_1 \propto x_2$, the target function $\tilde{f}(\alpha_1, \alpha_2)$ has a finite lower bound. Disregarding the constraint that $\alpha_i \in [0, C]$, I can minimize the target function at

$$\begin{aligned}
\alpha_2^* &= \frac{\gamma y_2 x_1^T x_1 - \gamma y_2 x_1^T x_2 - y_2(x_2^T - x_1^T) \sum_{i=3}^{N} \alpha_i y_i x_i - s + 1}{||x_1 - x_2||^2} \\
&= \frac{\alpha_2 ||x_1 - x_2||^2 + y_2 (x_1 - x_2)^T \beta - s + 1}{||x_1 - x_2||^2} \\
&= \alpha_2 + \frac{y_2 \beta^T (x_1 - x_2) - s + 1}{||x_1 - x_2||^2} \\
&= \alpha_2 + \frac{y_2(\beta^T x_1 - y_1) - y_2(\beta^T x_2 - y_2)}{||x_1 - x_2||^2} \\
&= \alpha_2 + y_2 \frac{E_1 - E_2}{||x_1 - x_2||^2},
\end{aligned} \tag{11}$$

where $E_1 = \beta^T x_1 + \beta_0 - y_1, E_2 = \beta^T x_2 + \beta_0 - y_2$ are the prediction errors. The right hand side of the equation depends only on the old α_2, and the left hand side of the equation is the new α_2. Thus, Equation [11] can be used to update the variation parameter α_2. Note that I also need to take constraint conditions into consideration. The constraints for α_1, α_2 are

$$\begin{aligned}
& 0 \leq \alpha_1, \alpha_2 \leq C, \\
& \alpha_1 y_1 + \alpha_2 y_2 = \gamma.
\end{aligned} \tag{12}$$

In order to determine the feasible range of α_1, α_2, I need to consider these exclusive cases:

Case I: $y_1 y_2 = -1$ Now the constraint is $\alpha_1 - \alpha_2 = k = \gamma y_1$. With some elementary algebra, the feasible ranges are determined as

$$\begin{aligned}
& \max(-k, 0) \leq \alpha_2 \leq C + \min(-k, 0) \\
& \max(-k, 0) \leq \alpha_2 \leq C + \min(-k, 0) \\
& -C \leq k \leq C
\end{aligned} \tag{13}$$

Case II: $y_1 y_2 = 1$ Now the constraint is $\alpha_1 + \alpha_2 = k = \gamma y_1$. With some algebra, the feasible range is found to be

$$\begin{aligned} &\max(0, k - C) \le \alpha_1 \le \min(C, k) \\ &\max(0, k - C) \le \alpha_2 \le \min(C, k) \\ &0 \le k \le 2C \end{aligned} \tag{14}$$

When the updated α_2^* lies within the feasible range, I will set the updated $\alpha_2^{\text{new}} = \alpha_2^*$; if α_2^* is smaller than the permitted lower bound of α_2, then $\alpha_2^{\text{new}} = \min(\alpha_2)$; when α_2^* is larger than the permitted upper bound of α_2, then $\alpha_2^{\text{new}} = \max(\alpha_2)$. Once we know the updated α_2^{new}, we can determine the updated α_1^{new} as $\alpha_1^{\text{new}} = \gamma y_1 - s\alpha_2^{\text{new}}$. If the updated α_2^{new} is calculated according the rules listed above, then it is guaranteed that the updated α_1^{new} also lies within the feasible range.

IV Program implementation and experimental results

I have written two programs to implement this algorithm. One in Python and another Scala. They both produce consistent results. The Python version of the program can be found here, and the Scala version of the program can be found here. I have tested the programs with synthetic two dimensional data, and the results are shown in Fig. IV.1.

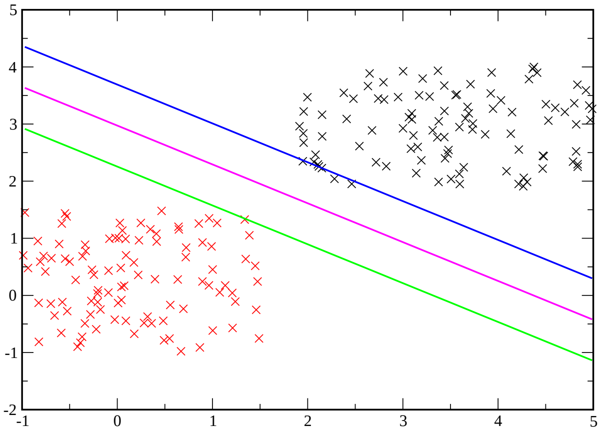

Figure IV.1: Experimental results of my SVM program. The points are linearly separated.