Constrained optimization: inequality constraints

Lluís Garrido – lluis.garrido@ub.edu

December 2016

Abstract

This laboratory is focused on constrained optimization and, in particular, on inequality constraints that appear in the support vector machines.

1 Support Vector Machines

Assume that we have a set of data points that we have *previously* classified in one of two ways: either they have a certain stated property or they do not. These data points might, for instance, represent the subject titles of email messages, which are classified as either being legitimate mail or spam. Suppose now that we obtain a new data point, i.e. a new email message. Our goal is to determine whether this new point does or does not have the stated property, i.e. if the email is legitimate or not. The set of techniques for doing this is broadly referred as pattern classification.

In its simplest form, pattern classification uses linear functions to provide the characterization. Suppose we have a set of m training data, $x_i \in R^n$, with classification y_i, where either $y_i = 1$ or $y_i = -1$ (the data point has a certain property or not), see Figure 1 on the left. Suppose it is possible to find some hyperplane $w^T x + b = 0$ which separates the positive points from the negative. For Figure 1 there is a hyperplane that clearly separates both classes. In this case

$$\begin{aligned} w^T x_i + b \geq +1 \qquad & \text{for} \qquad y_i = +1 \\ w^T x_i + b \leq -1 \qquad & \text{for} \qquad y_i = -1 \end{aligned}$$

We would like the hyperplanes separating the positive points from the negative to be as far apart as possible. From basic geometric principles it can be shown that the distance between the two hyperplanes (that is, the separation margin) is $2/\|w\|$. Thus among all separating hyperplanes we should seek the one that maximizes this margin. This is equivalent to minimizing $w^T w$. The resulting problem is to determine the coefficients w and b that solve

$$\begin{aligned} \text{mimize} \qquad & f(w,b) = \tfrac{1}{2} w^T w \\ \text{subject to} \qquad & y_i(w^T x_i + b) \geq 1 \qquad i = 1 \dots m \end{aligned}$$

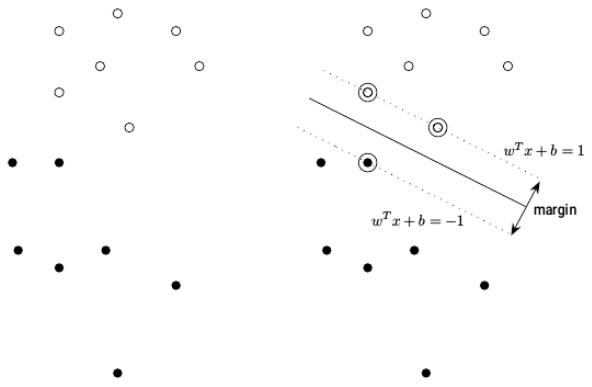

Figure 1: Linear separating hyperplane for the separable case. Image taken from Griva, I.; Nash, S.; Sofer, A., "Linear and nonlinear optimization", SIAM.

Once the coefficients w and b of the separating hyperplane are found from the training data, we can use the value of the function $f(x) = w^T x + b$ (our "learning machine") to predict whether a new point $\bar{x}$ has the property of interest or not, depending on the sign of $f(\bar{x})$.

So far we have assumed that the data set was separable, that is, a hyperplane separating the positive points from the negative points exists. For the case where the data set is not separable, we can refine the approach to the separable case, see Figure 2. We will now allow the points to violate the equations of the separating hyperplane, but we will impose a penalty for the violation. Letting the nonnegative variable ξ_i denote the amount by which the point x_i violates the constraint at the margin, we now require

$$\begin{aligned} w^T x_i + b &\geq +1 - \xi_i \qquad \text{for} \qquad y_i = +1 \\ w^T x_i + b &\leq -1 + \xi_i \qquad \text{for} \qquad y_i = -1 \end{aligned}$$

A common way to impose the penalty is to add to the objective a term proportional to the sum of the violations. The added penalty term takes the form $C \sum_{i=1}^{m} \xi_i$ and is added to the objective, where the larger the value of the parameter C, the larger the penalty for violating the separation. Our problem is now to find w, b and ξ that solve

$$\begin{array}{lll} \text{mimize} & f(w, b, \xi) = \frac{1}{2} w^T w + C \sum_{i=1}^{m} \xi_i & \\ \text{subject to} & y_i(w^T x_i + b) \geq 1 - \xi_i & i = 1 \ldots m \\ & \xi_i \geq 0 & \end{array} \tag{1}$$

This is the original or *primal* problem. How can this primal problem be solved? Many optimization problems have a companion problem called the dual problem. There are important relations between a primal and its dual, and that these relations sometimes lead to insights for solving the problem.

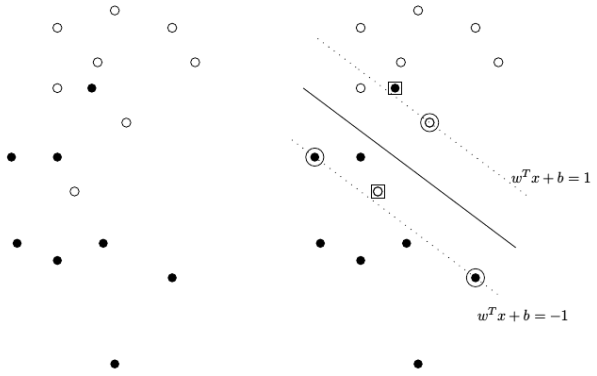

Figure 2: Linear separating hyperplane for the non separable case. Image taken from Griva, I.; Nash, S.; Sofer, A., "Linear and nonlinear optimization", SIAM.

The dual problem may be easier to solve, and if the optimal solution to the dual problem is known, then (in nondegenerate cases) the optimal solution to the primal problem can be easily computed.

Let us know formulate the dual of the primal Equationss (1). In the dual problem, the roles of variables and constraints are reversed with respect to the primal problem. That is, the variables in the original or primal program are "transformed" into constraints in the dual problem, and the constraints in the primal are "transformed" into variables in the dual problem. We are not going into all the mathematical details of the formulation presented next. If you are interested, just take a look e.g. at the book Griva, I.; Nash, S.; Sofer, A., "Linear and nonlinear optimization", SIAM.

To define the dual of this problem it will be convenient to represent the problem constraints in matrix-vector form. Denote by X the $n \times m$ matrix whose columns are the training vectors x_i. Let $Y = \text{diag}(y)$ be the diagonal matrix whose i-th diagonal term is y_i, and let $e = (1 \cdots 1)^T$ be a vector of length m whose entries are all equal to one. Then the primal problem can be rewritten as

$$\begin{array}{lll} \text{mimize} & f(w,b,\xi) = \frac{1}{2} w^T w + C e^T \xi & \\ \text{subject to} & Y X^T w + y b \geq e - \xi & i = 1 \ldots m \\ & \xi \geq 0 & \end{array} \tag{2}$$

Let α and η be the m-dimensional vectors of Lagrange multipliers corresponding to the first and second constraints, respectively, of the previous equation. The dual problem can be written just in terms of α

$$\begin{array}{ll} \text{maximize} & f(\alpha) = \sum_{i=1}^m \alpha_i - \frac{1}{2} \alpha^T (Y X^T X Y) \alpha \\ \text{subject to} & \sum_{i=1}^m y_i \alpha_i = 0 \\ & 0 \leq \alpha_i \leq C \end{array} \tag{3}$$

The dual, like the primal, is a quadratic problem. However, it is usually easier

to solve because, with the exception of one equality, all constraints are simple upper and lower bounds.

Assume that the dual problem has been solved, that is, we know the optimal values of the vector α. The parameters w and b are then computed as follows

$$w = XY\alpha = \sum_{i=1}^{m} \alpha_i y_i x_i$$

Any α_i that is strictly positive ($\alpha_i > 0$) corresponds indeed to a binding constraint and hence the corresponding point x_i is a support vector. Let us denote the set of support vectors by SV. The coefficients of the hyperplane w can then be computed as

$$w = \sum_{i \in \text{SV}} \alpha_i y_i x_i$$

A point x_j for which $0 < \alpha_j < C$ satisfies $\xi_j = 0$ and thus $y_j(w^T x_j + b) = 1$. Any such point j can be used to compute the value of b

$$b = y_j - w^T x_j$$

If there are several such points, the average computed value of b is commonly taken to ensure the highest accuracy.

In general, the significance of the dual formulation is its computational ease. But, for the support vector machine, it also has another important advantage: it allows us to expand the power of support vector machines to data that are not linearly separable. In other words, it allows to use other functions (called kernels) to separate data points. This is out of the scope of this laboratory.

2 Implementation of the dual formulation

We are interested in solving Equations (3) using gradient descent. Our purpose is to maximize the quadratic expression $f(\alpha) = \sum_{i=1}^{m} \alpha_i - \frac{1}{2}\alpha^T(YX^TXY)\alpha$ subject to the constraints $\sum_{i=1}^{m} y_i\alpha_i = 0$ and $0 \leq \alpha_i \leq C$. The first constraint is an equality constraint and the second an inequality. How can we deal with such constraints? We propose the next way to solve the dual formulation:

1. Assume that we only had to minimize $f(\alpha)$ without any constraints. In this case a simple gradient descent (or Newton, if you prefer) could be used. The descent would be

   $$\alpha^{k+1} = \alpha^k - \beta^k \nabla f(\alpha^k)$$

   where β^k is a step that can be computed using backtracking.

2. The constraint $\sum_{i=1}^{m} y_i\alpha_i = 0$ corresponds a hyperplane. The previous expression can be compactly expressed as $y^T\alpha = 0$, that is, y and α are orthogonal. Assume that α^k is on the hyperplane. We can ensure that α^{k+1} is on the hyperplane by projecting $\nabla f(\alpha^k)$ onto the hyperplane.

In order to do this, let $g = \nabla f(\alpha^k)$ the vector to be projected over the hyperplane defined by $y^T\alpha = 0$. We begin by first projecting g over y, $g^\perp = (g^T\hat{y})\hat{y}$, where $\hat{y} = y/\|y\|$. Then $g^\parallel = g - g^\perp$. Let us denote with $P^k = g^\parallel$ the corresponding projected gradient. Now, the descent over the hyperplane can be written as

$$\alpha^{k+1} = \alpha^k - \beta^k P^k$$

3. The last constraint $0 \leq \alpha_i \leq C$ might be implemented using interior or exterior penalty methods, for instance. Observe, however, that the constraint is directly imposed over the values of α_i we have to optimize. We can obtain the same effect as interior or exterior penalty methods with a combined projection of g over the previous hyperplane and the region defined by $0 \leq \alpha_i \leq C$. The projected vector D^k has to be inside the hyperplane and has to satisfy $0 \leq \alpha_i^k - \delta D^k \leq C$, where δ is an a small value.

   Figure 3 shows an example: assume that only one component i of g_i falls outside the cube, see Figure 3. In order to obtain D^k, we have to project the gradient $g = \nabla f(\alpha^k)$ over the intersection of the hyperplane and the plane defined by the i-th side of the cube.

   In the general case, D^k is the projection of g over the subspace orthogonal to y and $\{c_i\}$, where c_i is the set of faces at which g_i is "pointing" outside the cube. The upper or lower plane of the i-th face of the cube can be defined by its orthogonal vector $c_i = (0, 0, \ldots, 1, \ldots, 0)^T$. The vector c_i is a vector of zeros except at position i at which it is 1.The projection can then be obtained by constructing the subspace spanned by y and $\{c_i\}$ with the Gram-Schmidt algorithm. Once the subspace has been constructed, the orthogonal projection can be obtained in a similar way as P^k has been obtained in step 2, see python code that is included with this document.

   Once D^k is obtained, it is recommended (due to numerical precision) to manually set to zero the i components of D^k at which g_i "points" outside the cube.

4. It should be noted that $f(\alpha)$ is a quadratic form. Thus, the value of β that minimizes $f(\alpha^k - \beta D^k)$ has a closed form! Which is this closed form? Assume we want to minimize

$$\min f(\alpha) = \frac{1}{2}\alpha^T A\alpha - h^T\alpha$$

   where A is assumed to be positive definite and symmetric (in our problem, for the support vector machine, $A = YX^TXY$ and $h = e$). Given an initial point α_0 and a descent direction p (in our problem $p = -D^k$), the function $f(\alpha_0 + \beta p)$ is minimized for

$$\beta_{\text{opt}} = \frac{(h - A\alpha_0)^T p}{p^T A p}$$

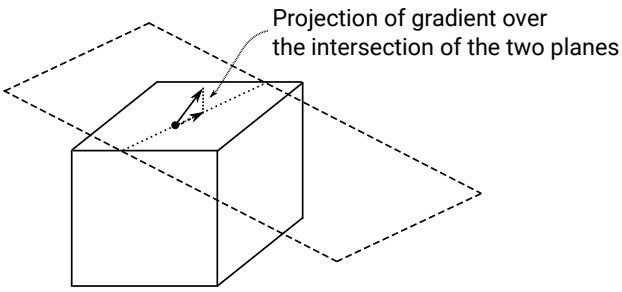

Figure 3: Projection of a gradient vector g over the hyperplane and one of the sides of the cube.

Once β has been computed, you may compute the maximum value of β, $\beta_{\max}$, so as to ensure that $0 \leq \alpha^k - \beta D^k \leq C$. If $\beta_{\text{opt}} > \beta_{\max}$, you will have to take $\beta = \beta_{\max}$ to update the values of α. In addition, you will be sure that you will arrive to at least one side of the cube. If, on the other hand, $\beta_{\text{opt}} < \beta_{\max}$, you may take $\beta = \beta_{\text{opt}}$ to update the values of α and you won't arrive to any side of the cube. The value of α is updated with

$$\alpha^{k+1} = \alpha^k - \beta^k D^k$$

Report

What are you expected to do? Implement the previous proposed algorithm and check if the solutions you obtain are similar to those obtained with the ones obtained in the lectures of Oriol. You are recommended to begin with the simplest case, i.e. the case in which data is separable. For that issue, use a very high value of C. In addition, it is recommended to use a small dataset (4 points, for instance). Once it works, you may test your algorithm with the non-separable case. But take into account that numerical details may lead to difficulties in the latter case.

You are asked to deliver an *individual* report of the exercise proposed in section 2. Just explain each of the steps you have followed. The deadline to deliver this report is January 10th at 9 p.m. (21h).

Note

Most of the text of section 1 has been copied from different chapters of Griva, I.; Nash, S.; Sofer, A., "Linear and nonlinear optimization", SIAM.