9.520: Class 22

Current Topics of Research III: Theory And Implmentation Of Support Vector Machines

Ryan Rifkin

About this class

Theme Support Vector Machines are discussed in detail. The connection between the primal and dual forms is explored. Alternate forms of the SVM problem are formed. How to solve the standard SVM problem is discussed.

Math required Lagrange multipliers technique.

Web The slides and all what concerns this class can be found on the web.

Plan

- Derivation of the SVM Dual
- Alternate Forms of SVMs
- How To Solve SVMs

The Primal and Dual Formulations

We restate the primal and dual formulations for linear SVMs:

$$\text{(P)} \quad \min_{\mathbf{w}, b, \xi} \quad \frac{1}{2}\|\mathbf{w}\|^2 + C\sum \xi_i$$
$$y_i(\mathbf{w}\cdot\mathbf{x}_i + b) \geq 1 - \xi_i$$
$$\xi_i \geq 0$$

$$\text{(D)} \quad \max_{\alpha} \quad \alpha\cdot\mathbf{1} - \frac{1}{2}\alpha\mathbf{D}\alpha$$
$$\alpha\cdot\mathbf{y} = 0$$
$$0 \leq \alpha \leq C$$

where $D_{ij} \equiv y_i y_j \mathbf{x}_i \cdot \mathbf{x}_j$. What is the connection between (P) and (D)?

Motivation From Regularization Theory, I

To perform structural risk minimization, we'd like to solve:

$$\text{(SRM)} \quad \min_{f} \quad \sum V(f(\mathbf{x_i}, y_i)) \qquad \|f\| \leq A$$

for a variety of values of A, and then pick the value of A that minimizes a bound on generalization error.

If our loss function V is the hinged loss

$$V(f(\mathbf{x_i}, y_i)) = (1 - y_i f(\mathbf{x_i}))_+$$

then we can rewrite the problem as:

Motivation From Regularization Theory, II

$$\begin{array}{lll} \text{(R)} & \min\limits_{\mathbf{w}, b, \xi} & \sum \xi_i \\ & & y_i(\mathbf{w} \cdot \mathbf{x}_i + b) \geq 1 - \xi_i \\ & & \|w\| \leq A \\ & & \xi_i \geq 0 \end{array}$$

Our actual procedure differs from this in two (one and a half??) ways:

Motivation From Regularization Theory, III

- We choose C, not A, optimizing $\frac{1}{2}\|w\|^2 + C\sum\xi_i$. A choice of A is **implicit** here, and is **data dependent**. Increasing C is equivalent to increasing A, although the precise relationship is unknown.

- In practice, we often use a **single** value of C, moving further away from structural risk minimization.

The Primal Formulation, w/Lagrange Multipliers

We will start with the primal problem, and using the technique of Lagrange multipliers, derive the dual. We begin by associating Lagrange multiplier variables with each primal constraint:

$$
\begin{array}{lll}
\text{(P)} \quad \min\limits_{\mathbf{w}, b, \xi} & \frac{1}{2}\|\mathbf{w}\|^2 + C \sum \xi_i & \\
& y_i(\mathbf{w} \cdot \mathbf{x}_i + b) \geq 1 - \xi_i & (\alpha_i) \\
& \xi_i \geq 0 & (\mu_i)
\end{array}
$$

The Lagrangian

This leads to a formula called the Lagrangian:

$$L(\mathbf{w}, b, \xi, \alpha, \mu) = \\ \frac{1}{2}\|\mathbf{w}\|^2 + C\sum \xi_i - \sum \alpha_i (y_i(\mathbf{w} \cdot \mathbf{x}_i + b) - 1 + \xi_i) - \sum \mu_i \xi_i$$

We find our solution by simultaneously **minimizing L** with respect to our original primal variables $\mathbf{w}, b$, and ξ, and **maximizing L** with respect to our new, dual variables α and μ, subject to the constraints that $\alpha \geq 0$ and $\mu \geq 0$.

Removing b and ξ, I

Taking derivatives with respect to b and ξ, we find:

$$\frac{\partial L}{\partial b} = 0 \;\; \Rightarrow \;\; \sum \alpha_i y_i = 0$$

$$\frac{\partial L}{\partial \xi_i} = 0 \;\; \Rightarrow \;\; C - \alpha_i - \mu_i = 0$$

Removing b and ξ, II

Substituting back into L, we obtain the new Lagrangian:

$$L'(\mathbf{w}, \alpha, \mu) = \frac{1}{2}\|\mathbf{w}\|^2 - \sum \alpha_i (y_i(\mathbf{w} \cdot \mathbf{x_i}) + 1)$$

which we want to minimize with respect to $\mathbf{w}$, and maximize with espect to α and μ, subject to the constraints:

$$\begin{aligned} \alpha &\geq \mathbf{0} \\ \mu &\geq \mathbf{0} \\ \sum \alpha_i y_i &= 0 \\ C - \alpha_i - \mu_i &= 0 \\ \alpha_i, \mu_i &\geq 0 \end{aligned}$$

Removing w

$$\frac{\partial L'}{\partial \mathbf{w}} = 0 \Rightarrow w - \sum \alpha_i y_i \mathbf{x_i} = 0 \Rightarrow w = \sum \alpha_i y_i \mathbf{x_i}$$

The optimal hyperplane $\mathbf{w}$ can be expressed as a sum of the input vectors $\mathbf{x_i}$, multiplied by their class y_i and their dual variables α_i. We also knew this directly from regularization theory in a previous class.

Making this substitution, noting that

$$(\sum y_i \alpha_i \mathbf{x_i})^2 = \sum \sum y_i y_j \alpha_i \alpha_j \mathbf{x_i} \cdot \mathbf{x_j},$$

and noting that after this substitution, we are maximizing with respect to all remaining variables, we obtain the following dual program:

Almost The Dual Program

$$
\begin{aligned}
(\mathrm{D}') \quad \max_{\alpha,\mu} \quad & \alpha \cdot \mathbf{1} - \tfrac{1}{2}\alpha \mathbf{D} \alpha \\
\alpha \cdot \mathbf{y} &= 0 \\
\mathbf{C} - \alpha - \mu &= 0 \\
\alpha &\geq 0 \\
\mu &\geq 0
\end{aligned}
$$

where $D_{ij} \equiv y_i y_j \mathbf{x_i} \cdot \mathbf{x_j}$. Because μ does not appear in the objective function, we can remove it from the final formulation, leading to the SVM dual...

The SVM Dual Program

$$\begin{aligned} \text{(D)} \quad \max_{\alpha} \quad & \alpha \cdot \mathbf{1} - \tfrac{1}{2}\alpha \mathbf{D} \alpha \\ & \alpha \cdot \mathbf{y} = 0 \\ & \alpha \geq 0 \\ & \alpha \leq \mathbf{C} \end{aligned}$$

Nonlinear SVMs, I

The "traditional" approach is to start with linear SVMs, derive the dual, then note that in the dual, the $\mathbf{x_i}$ vectors themselves are not required, but only dot products of the form $\mathbf{x_i} \cdot \mathbf{x_j}$. We substitute $K(\mathbf{x_i}, \mathbf{x_j}) \equiv K_{ij} \equiv \Phi(\mathbf{x_i}) \cdot \Phi(\mathbf{x_j})$, which gives a different D matrix. but the same quadratic program formulation.

If, instead, we address nonlinear SVMs by **assuming the form** of the solution:

$$\mathbf{w} \equiv \sum y_i \gamma_i \Phi(\mathbf{x_i})$$

we can derive both primal and dual nonlinear SVM formulations. The derived programs are...

Primal and Dual Nonlinear SVMs

$$
\begin{array}{lll}
\text{(PNL)} & \min\limits_{\gamma, b, \xi} & \frac{1}{2}\gamma \mathbf{D} \gamma + C \sum \xi_i \\
& & y_i(\sum y_j \gamma_j K_{ij} + b) \geq 1 - \xi_i \\
& & \xi_i \geq 0
\end{array}
$$

$$
\begin{array}{lll}
\text{(DNL)} & \max\limits_{\alpha} & \alpha \cdot \mathbf{1} - \frac{1}{2}\alpha \mathbf{D} \alpha \\
& & \alpha \cdot \mathbf{y} = 0 \\
& & 0 \leq \alpha \leq C
\end{array}
$$

Convexity

If K is a valid kernel (i.e., K satisfies Mercer's Condition), the matrix D is **positive semidefinite**, and the problems **PNL** and **DNL** are both feasible convex quadratic programs. Therefore, they both have optimal solutions, and those optimal solutions will have the **same** objective values. The optimal solutions are not necessarily unique, unless the matrix D is **positive definite**.

Complementary Slackness

We can relate solutions to the primal and dual problems using **complementary slackness**. For our purposes, we can interpret this to mean this if we have an **inequality** constraint $g(x) \leq 0$, and an associated dual variable k, then, if we have **optimal** solutions to the primal and dual problems, at optimality, $k \cdot g(x) = 0$. In other words, either the dual variable is 0 or the inequality is satisfied as an equality.

Using complementary slackness, we see that if γ, b, and ξ are an optimal solution to **PNL**, and α is an optimal solution to **DNL**, then all of the following hold:

Conditions Holding At Optimality

$$
\begin{aligned}
D\gamma - D\alpha &= 0 && (1)\\
\sum \alpha_i y_i &= 0 && (2)\\
C - \alpha_i - \mu_i &= 0 && (3)\\
y_i(\sum y_j \gamma_j K_{ij} + b) - 1 + \xi_i &\geq 0 && (4)\\
\alpha_i\{y_i(\sum y_j \gamma_j K_{ij} + b) - 1 + \xi_i\} &= 0 && (5)\\
\mu_i \xi_i &= 0 && (6)\\
\xi_i, \alpha_i, \mu_i &\geq 0 && (7)
\end{aligned}
$$

From Dual To Primal

Suppose α solves **DNL**. Then we can easily use the optimality conditions to get a solution to **PNL**. First, we set $\gamma = \alpha$. Next, we assume that for some i, $0 \leq \alpha_i \leq C$ (if not, we are in trouble). By Eqn. 5:

$$y_i(\sum y_j \gamma_j K_{ij} + b) - 1 + \xi_i\} = 0$$

Also, $\mu_i > 0$ by Eqn. 3, so $\xi_i = 0$ by Eqn. 6. We conclude that b can determined by:

$$b = y_i - \sum y_j \gamma_j K_{ij}$$

Once we have b, the ξ_i can be determined from Eqn. 4.

From Primal To Dual?

Given a solution to **PNL**, we cannot necessarily determine a solution to **DNL**. We try to set $\alpha = \gamma$ to satisfy Eqn. 1, but this choice of α may not satisfy Eqn. 2. Even though every valid α is a valid γ, a valid γ may not be a valid α.

If the matrix D is *positive definite*, then it is possible to show that the difficulty disappears, and that optimal γ and α exist, and are unique.

This should be viewed as the more "usual" case. Of course, the primal solution is what we actually **want**...

Why Is Training an SVM Hard?

Suppose you are given a "black box" which solves QP problems of the form **PNL** or **DNL**. Such packages exist — CPLEX, MINOS, LOQO, Matlab QP Solver, etc.

The problem: if we have n data points, we need $O(n^2)$ memory just to write down the matrix D. If $n = 20000$, and it takes 4 bytes to represent an entry of D, we would need 1.6 Gigabytes to store the D matrix.

So we can train small SVMs using an arbitrary QP package, but this doesn't scale to large problems. What do we do?

The Answer: Decomposition

We can solve a large SVM dual problem by solving a sequence of smaller problems. Suppose we divide α into two sets, a **working set** α_W and the remaining variables α_R. We can rewrite the dual problem:

$$\begin{array}{lll} \text{(DNL)} & \max\limits_{\alpha} & [\alpha_{\mathbf{W}}\alpha_{\mathbf{R}}] \cdot \mathbf{1} - \frac{1}{2}[\alpha_{\mathbf{W}}\alpha_{\mathbf{R}}] \begin{bmatrix} D_{WW} & D_{WR} \\ D_{RW} & D_{RR} \end{bmatrix} \begin{bmatrix} \alpha_{\mathbf{W}} \\ \alpha_{\mathbf{R}} \end{bmatrix} \\ & & [\alpha_{\mathbf{W}}\alpha_{\mathbf{R}}] \cdot \mathbf{y} = 0 \\ & & 0 \leq \alpha \leq C \end{array}$$

The Reduced Problem

Suppose we are given some feasible α. Now, we view α_W as variables and α_R as **constants**. Suppressing constant terms, we can rewrite our problem as:

$$\begin{array}{lll} \text{(DNLW)} & \max\limits_{\boldsymbol{\alpha_W}} & (\mathbf{1} - \mathbf{D_{WR}}\boldsymbol{\alpha_R})\boldsymbol{\alpha_W} - \frac{1}{2}\boldsymbol{\alpha_W}\mathbf{D_{WW}}\boldsymbol{\alpha_W} \\ & & \boldsymbol{\alpha_W} \cdot \mathbf{y_W} = -\boldsymbol{\alpha_R} \cdot \mathbf{y_R} \\ & & 0 \leq \alpha_W \leq C \end{array}$$

DNLW has the **same form** as **DNL**, and can be solved using a black box.

Decomposition Cont'd.

The basic method: at each iteration, choose a **working set** α_W, and optimize just those variables by solving **DNLW**.

When we optimize over α_W, holding α_R constant, we must move towards the optimum over all of α — those terms that involve α_R only remain constant, and all other terms are optimized over:

$$\mathbf{Obj(DNL)} = \mathbf{Obj(DNLW)} + \mathbf{F}(\alpha_\mathbf{R})$$

Choosing A Working Set

Looking back at our optimality conditions, and assuming that optimality, we'll set $\gamma = \alpha$, we will have:

$$\begin{array}{rcl} \alpha_i = 0 & \Longrightarrow & \sum_{y_j \alpha_j K_{ij}} + b \geq 1 \\ 0 \leq \alpha_i \leq C & \Longrightarrow & \sum_{y_j \alpha_j K_{ij}} + b = 1 \\ \alpha_i = C & \Longrightarrow & \sum_{y_j \alpha_j K_{ij}} + b \leq 1 \end{array}$$

Traditionally, at any given time, one "guesses" that one's α is correct, uses a α_i that is between 0 and C to determine a provisional b, then finds α_i which violate the above optimality conditions. Violating points are added.

It is possible to work directly with the gradient of **DNL** without computing an intermediate b.

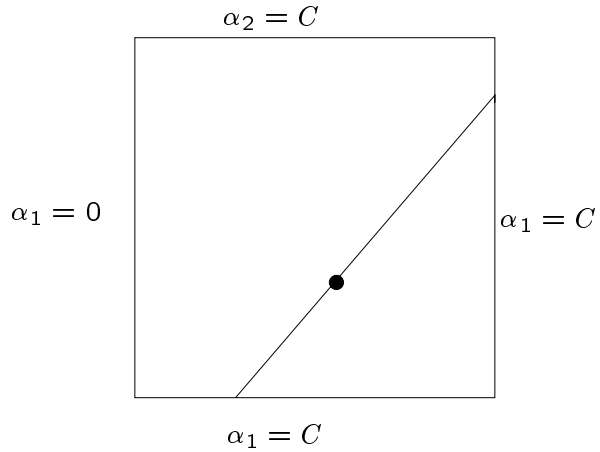
Solving Large-Scale SVMs

We now have a method for solving large SVM problems with a fixed amount of memory, assuming we have a method for solving "reasonable sized" problems of the form **DNL**. These packages exist, but the best ones are quite expensive, and they sometimes have numerical convergence issues...

Is there another way?

Optimizing Two Points At A Time

If we require the working set to be of size 2, the resulting subproblem can be solved analytically — we are simply minimizing a quadratic function of two variables, subject to an equality constraint and box constraints.



Why Is Training an SVM Slow?

Computing a single kernel product K_{ij} requires $O(n)$ time, where n is the input dimensionality.

It is crucial to **cache** kernel products that are going to be needed again. This is what makes the problem difficult and complex. Differences in caching strategies and performances lead to most of the differences between real-world algorithms.

SvmFu

SvmFu, a package developed at CBCL a mixed-level strategy:

- Breaks the problem up into medium-large subproblems (user-defined size)
- Large subproblems are directly related to the amount of caching
- Large subproblems are themselves solved by optimizing two elements at a time
- http://fpn.mit.edu/SvmFu

Linear SVMs Can Be Much Faster

To check whether a given point x_i, violates optimality conditions, we need to compute $f(x_i)$. In the nonlinear case, we need to compute

$$\sum \alpha_j y_j K(x_i, x_j) + b.$$

In the linear case, if we maintain w, we only need to compute

$$w \cdot x + b,$$

which can be **much** faster.

This is especially true in the **testing** stage.

b In The Objective Function, I

We can add $\frac{1}{2}b^2$ to the objective function:

$$\begin{array}{lll} \text{(PNLB)} \quad \min\limits_{\gamma, b, \xi} & \frac{1}{2}(\gamma \mathbf{D} \gamma + b^2) + C \sum \xi_i & \\ & y_i(\sum y_j \gamma_j K_{ij} + b) & \geq \quad 1 - \xi_i \\ & \xi_i & \geq \quad 0 \end{array}$$

b In The Objective Function, II

Geometrically, this means that we are now penalizing the minimum distance between the hyperplane and the origin. We "expect" the data to be zero-mean, and penalize deviations from this.

The Lagrangian becomes:

$$L(\mathbf{w}, b, \xi, \alpha, \mu) = \\ \frac{1}{2}\|\mathbf{w}\|^2 + \frac{1}{2}b^2 + C\sum\xi_i - \sum\alpha_i(y_i(\mathbf{w}\cdot\mathbf{x}_i + b) - 1 + \xi_i) - \sum\mu_i\xi_i$$

b In The Objective Function, III

Taking the derivative of this new Lagrangian with respect to b:

$$\frac{\partial L}{\partial b} = 0 \;\; \Rightarrow \;\; b - \sum \alpha_i y_i = 0$$

Other than this, everything else is the same, and the new dual is:

$$\begin{array}{lll} \text{(DNLB)} & \max\limits_{\alpha} & \alpha \cdot \mathbf{1} - \frac{1}{2}\alpha \mathbf{D} \alpha \\ & & 0 \leq \alpha \;\; \leq \;\; C \end{array}$$

If we solve $\mathbf{DNLB}$, we no longer need an equality constraint (making the problem **somewhat** easier), and at the end, we set $b = \sum y_i \alpha_i$.

No b at All

We can remove b entirely, forcing the hyperplane to go through the origin, resulting in the following pair of problems:

$$\begin{array}{lll} \text{(PNLNB)} & \min\limits_{\gamma, b, \xi} & \frac{1}{2}\gamma \mathbf{D} \gamma + C \sum \xi_i \\ & & y_i(\sum y_j \gamma_j K_{ij}) \geq 1 - \xi_i \\ & & \xi_i \geq 0 \end{array}$$

$$\begin{array}{lll} \text{(DNLNB)} & \max\limits_{\alpha} & \alpha \cdot \mathbf{1} - \frac{1}{2}\alpha \mathbf{D} \alpha \\ & & 0 \leq \alpha \leq C \end{array}$$

Sparsity Control, I

Imagine we are solving the primal problem, but we're getting too many support vectors. We could try to penalize the number of SVs directly:

$$\begin{array}{lll} \text{(PNLL0)} & \min\limits_{\gamma, b, \xi} & \frac{1}{2}\gamma \mathbf{D} \gamma + C \sum \xi_i + D\|\gamma\|_{L0} \\ & & y_i(\sum y_j \gamma_j K_{ij}) \geq 1 - \xi_i \\ & & \xi_i \geq 0 \end{array}$$

where, $\|\gamma\|_{L0}$ is the number of non-zero entries in γ. Unfortunately, this is an intractable, NP-hard optimization problem.

Sparsity Control, II

We **can**, however, penalize γ in the L_1 norm, which can be shown to have a somewhat similar effect:

$$\begin{array}{lll} \text{(PNLL1)} & \min\limits_{\gamma, b, \xi} & \frac{1}{2}\gamma \mathbf{D} \gamma + C \sum \xi_i + D\|\gamma\|_{L1} \\ & & y_i(\sum y_j \gamma_j K_{ij}) \geq 1 - \xi_i \\ & & \xi_i \geq 0 \end{array}$$

Several variants of this are possible. Unfortunately, none of them seem to lead to formulations with duals that decompose, so they can only be used on relatively small problems.