SVM optimization problem and its dual

Daniel Hsu (COMS 4771)

In this note, we derive the dual form of the SVM optimization problem. The dual form reveals some interesting properties of SVM, and also lends itself to the "kernel" version of SVMs.

SVM optimization problem

Suppose the data set $S = \{(\vec{x}_i, y_i)\}_{i=1}^n$, where $\vec{x}_i \in \mathbb{R}^d$ and $y_i \in \{0, 1\}$ for each $i = 1, \ldots, n$, is linearly separable. The SVM optimization problem characterizes the weight vector for the linear classifier that maximizes the worst margin achieved on S.

As described in lecture, the SVM optimization problem asks for the shortest weight vector $\vec{w} \in \mathbb{R}^d$ that satisfies the following two properties:

1. If $y_i = 1$, then $\vec{x}_i \cdot \vec{w} > 0$; if $y_i = 0$, then $\vec{x}_i \cdot \vec{w} < 0$.
2. $|\vec{x}_i \cdot \vec{w}| \geq 1$ for all $i = 1, \ldots, n$.

(We can ignore what happens when $\vec{x}_i \cdot \vec{w} = 0$ since it is precluded by the second property anyway.)

The first property says that the linear classifier corresponding to $\vec{w}$ should be a linear separator. The second property ensures that maximizing $1/\|\vec{w}\|_2$ (which is the same as minimizing $\|\vec{w}\|_2$) has the semantics of maximizing the worst margin achieved by the linear classifier corresponding to $\vec{w}$.[1]

Standard form of the SVM optimization problem

Let us change the label 0 to -1, so each $y_i \in \{-1, 1\}$. Now the first property from above becomes

1. If $y_i = 1$, then $\vec{x}_i \cdot \vec{w} > 0$; if $y_i = -1$, then $\vec{x}_i \cdot \vec{w} < 0$.

If the first property holds, then the second property can be re-written as

2. $y_i(\vec{x}_i \cdot \vec{w}) \geq 1$ for all $i = 1, \ldots, n$.

[1] In lecture, this second property was written as $\min_{i \in \{1,\ldots,n\}} |\vec{x}_i \cdot \vec{w}| = 1$. But it is clear that if you are seeking the minimum length $\vec{w}$ that satisfies $|\vec{x}_i \cdot \vec{w}| \geq 1$ for all $i = 1, \ldots, n$, you will indeed choose a weight vector $\vec{w}$ for which the minimum such value $|\vec{x}_i \cdot \vec{w}|$ is equal to 1.

In fact, if this new version of the second property holds, then the first property also holds. So we can replace the two properties in the SVM optimization problem with just this second one.

We've simplified (somewhat) the SVM optimization problem to the following: Find the shortest weight vector $\vec{w} \in \mathbb{R}^d$ that satisfies $y_i(\vec{x}_i \cdot \vec{w}) \geq 1$ for all $i = 1, \ldots, n$.

And when we say "shortest", we can also say "of minimum half squared Euclidean length" — we'll do this purely for mathematical convenience. So the final "standard form" of the optimization problem is:

$$\begin{aligned} \min_{\vec{w} \in \mathbb{R}^d} \quad & \frac{1}{2}\|\vec{w}\|_2^2 \\ \text{s.t.} \quad & y_i(\vec{x}_i \cdot \vec{w}) \geq 1 \quad \text{for all } i = 1, \ldots, n. \end{aligned}$$

Here, we have a constrained optimization problem over the d optimization variables $\vec{w} = (w_1, \ldots, w_d)$. The objective function is $\frac{1}{2}\|\vec{w}\|_2^2$, and there are n inequality constraints, one per training example.

This standard form of the SVM optimization problem is often referred to as the "primal" SVM optimization problem, which is foreshadowing the introduction of the "dual" SVM optimization problem, discussed shortly.

A two-player zero-sum game and the Lagrangian

The first step to obtaining the dual form of the SVM optimization problem is to use the method of Lagrange multipliers. We define a new "objective" function that incorporates both the original (primal) objective function and the constraints. The new function, called the Lagrangian, is a function of both the original (primal) optimization variables $\vec{w}$, as well as n additional variables $\vec{\lambda} = (\lambda_1, \ldots, \lambda_n)$, called Lagrange multipliers. There is one Lagrange multiplier per constraint. The Lagrangian function is

$$L(\vec{w}, \vec{\lambda}) = \frac{1}{2}\|\vec{w}\|_2^2 + \sum_{i=1}^{n} \lambda_i(1 - y_i(\vec{x}_i \cdot \vec{w})).$$

The Lagrangian function can be thought of as the "payoff value" of a zero-sum game between two players: an Optimizer, whose goal is to choose $\vec{w}$ so as to minimize the payoff value; and an Adversary, whose goal is to choose $\vec{\lambda}$ so as to maximize the payoff value. (Think of the payoff value as the amount that the Optimizer has to pay to the Adversary.) The Optimizer is allowed to choose any $\vec{w} \in \mathbb{R}^d$, while the Adversary is only allowed to choose $\vec{\lambda} \geq \vec{0}$ (i.e., non-negative values). In this game, the Optimizer has to choose $\vec{w}$ first; the Adversary is allowed to pick $\vec{\lambda}$ after seeing the choice of the Optimizer. If both players make their choices optimally, the payoff value is

$$\min_{\vec{w} \in \mathbb{R}^d} \max_{\vec{\lambda} \geq \vec{0}} L(\vec{w}, \vec{\lambda}).$$

Since the Adversary gets to choose $\vec{\lambda}$ after having seen the choice of the Optimizer, we can try to reason about the Adversary's "best response" to the Optimizer's choice. First, observe that the Adversary can only affect the term $\sum_{i=1}^{n} \lambda_i(1 - y_i(\vec{x}_i \cdot \vec{w}))$ in the Lagrangian. If the Optimizer chooses $\vec{w}$ that satisfies all of the constraints, then every value of $1 - y_i(\vec{x}_i \cdot \vec{w})$ is non-positive. If $1 - y_i(\vec{x}_i \cdot \vec{w}) < 0$, then choosing λ_i to be a positive number only helps the Optimizer, so the Adversary will choose $\lambda_i = 0$ for such cases. (If $1 - y_i(\vec{x}_i \cdot \vec{w}) = 0$, then the value of λ_i doesn't matter at all; it could be positive or zero.) So, the upshot is that if the Optimizer chooses $\vec{w}$ that satisfies all of the constraints, then the best response of the Adversary will be to choose $\vec{\lambda}$ so that $\sum_{i=1}^{n} \lambda_i(1 - y_i(\vec{x}_i \cdot \vec{w})) = 0$. This means that for any $\vec{w}$ that satisfies all of the constraints,

$$\max_{\vec{\lambda} \geq \vec{0}} L(\vec{w}, \vec{\lambda}) = \frac{1}{2}\|\vec{w}\|_2^2.$$

On the other hand, if the Optimizer chooses $\vec{w}$ that violates one of the constraints—say, the i-th constraint—then $1 - y_i(\vec{x}_i \cdot \vec{w}) > 0$. In this case, the Adversary can capitalize on this violation by choosing $\lambda_i \to \infty$ (i.e., choose λ_i to be an arbitrarily large positive number), upon which the payoff value also goes to infinity. This means that for any $\vec{w}$ that violates at least one of the constraints,

$$\max_{\vec{\lambda} \geq \vec{0}} L(\vec{w}, \vec{\lambda}) = +\infty.$$

To summarize, we've argued that

$$\max_{\vec{\lambda} \geq \vec{0}} L(\vec{w}, \vec{\lambda}) = \begin{cases} \frac{1}{2}\|\vec{w}\|_2^2 & \text{if } \vec{w} \text{ satisfies all constraints,} \\ +\infty & \text{otherwise.} \end{cases}$$

So minimizing $\max_{\vec{\lambda} \geq \vec{0}} L(\vec{w}, \vec{\lambda})$ is the same as minimizing the primal objective function subject to the constraints. The objective of the Optimizer in the game is, indeed, aligned with our goal of solving the SVM optimization problem.

Weak duality

In two-player zero-sum games, such as the game we just discussed, the player who chooses second has an advantage. In our "Adversary-chooses-second" game, the Adversary is able to make the payoff at least as large as what is possible in the "Optimizer-chooses-second" game. This simple observation can be expressed as

$$\min_{\vec{w} \in \mathbb{R}^d} \max_{\vec{\lambda} \geq \vec{0}} L(\vec{w}, \vec{\lambda}) \geq \max_{\vec{\lambda} \geq \vec{0}} \min_{\vec{w} \in \mathbb{R}^d} L(\vec{w}, \vec{\lambda}).$$

This is a property called *weak duality*, and it is a general property about such two-player zero-sum games (or equivalently, the optimization problems from which the games are derived).

Strong duality

It is somewhat of a miracle that for the SVM optimization problem, the optimal payoff value is the same in both the "Adversary-chooses-second" and the "Optimizer-chooses-second" game. In other words, choosing second does not actually give an advantage. This miracle is called *strong duality*, and we can write its consequence mathematically as

$$\min_{\vec{w} \in \mathbb{R}^d} \max_{\vec{\lambda} \geq \vec{0}} L(\vec{w}, \vec{\lambda}) = \max_{\vec{\lambda} \geq \vec{0}} \min_{\vec{w} \in \mathbb{R}^d} L(\vec{w}, \vec{\lambda}).$$

There are two nice things about the "Optimizer-chooses-second" game. First, the goal of the Optimizer is to minimize $J_{\vec{\lambda}}(\vec{w}) := L(\vec{w}, \vec{\lambda})$, and there are no constraints on the allowed values for the Optimizer. Second, the minimizer of $J_{\vec{\lambda}}$ can be easily deduced using calculus, because $J_{\vec{\lambda}}(\vec{w})$ is a convex quadratic function of $\vec{w}$, and it is easy to analytically decude the minimizers of such functions. The gradient of $J_{\vec{\lambda}}$ with respect to $\vec{w}$ is

$$\frac{\partial J_{\vec{\lambda}}}{\partial \vec{w}}(\vec{w}) = \vec{w} - \sum_{i=1}^{n} \lambda_i y_i \vec{x}_i.$$

The gradient is $\vec{0}$ if and only if $\vec{w} = \sum_{i=1}^{n} \lambda_i y_i \vec{x}_i$. And because $J_{\vec{\lambda}}(\vec{w})$ is convex, this value of $\vec{w}$ must be the minimizer of the function.

So, for a fixed value of $\vec{\lambda}$, the Lagrangian is minimized (with respect to the setting of $\vec{w}$) by choosing $\vec{w} = \sum_{i=1}^{n} \lambda_i y_i \vec{x}_i$. This is the Optimizer's best response to the Adversary's choice.

An important observation we see is that the Optimizer's best response is a linear combination of the $\vec{x}_i$'s. This means that the solution to the SVM optimization problem is in the span of the $\vec{x}_i$'s.

Dual SVM optimization problem

Let us recall the form of the Lagrangian and re-write in a slightly different way:

$$\begin{aligned} L(\vec{w}, \vec{\lambda}) &= \frac{1}{2}\|\vec{w}\|_2^2 + \sum_{i=1}^{n} \lambda_i (1 - y_i(\vec{x}_i \cdot \vec{w})) \\ &= \frac{1}{2}\vec{w} \cdot \vec{w} + \sum_{i=1}^{n} \lambda_i - \left(\sum_{i=1}^{n} \lambda_i y_i \vec{x}_i \right) \cdot \vec{w}. \end{aligned}$$

Now plug-in the Optimizer's best response for the variable $\vec{w}$ in the Lagrangian. The resulting function of $\vec{\lambda}$ is

$$\begin{aligned}
&L\left(\sum_{i=1}^{n} \lambda_i y_i \vec{x}_i, \vec{\lambda}\right) \\
&= \frac{1}{2}\left(\sum_{i=1}^{n} \lambda_i y_i \vec{x}_i\right) \cdot \left(\sum_{i=1}^{n} \lambda_i y_i \vec{x}_i\right) + \sum_{i=1}^{n} \lambda_i - \left(\sum_{i=1}^{n} \lambda_i y_i \vec{x}_i\right) \cdot \left(\sum_{i=1}^{n} \lambda_i y_i \vec{x}_i\right) \\
&= -\frac{1}{2}\left(\sum_{i=1}^{n} \lambda_i y_i \vec{x}_i\right) \cdot \left(\sum_{i=1}^{n} \lambda_i y_i \vec{x}_i\right) + \sum_{i=1}^{n} \lambda_i \\
&= -\frac{1}{2}\sum_{i=1}^{n}\sum_{j=1}^{n} \lambda_i \lambda_j y_i y_j (\vec{x}_i \cdot \vec{x}_j) + \sum_{i=1}^{n} \lambda_i .
\end{aligned}$$

The "dual" SVM optimization problem is to maximize this function of $\vec{\lambda}$ over $\vec{\lambda} \geq \vec{0}$:

$$\max_{\vec{\lambda} \geq \vec{0}} \quad -\frac{1}{2}\sum_{i=1}^{n}\sum_{j=1}^{n} \lambda_i \lambda_j y_i y_j (\vec{x}_i \cdot \vec{x}_j) + \sum_{i=1}^{n} \lambda_i .$$

Solving this problem yields the optimal settings of the Lagrange multipliers $\vec{\lambda}$. Plugging these values into the dual objective gives the optimal value of the primal SVM optimization problem. To get the value of $\vec{w}$ that solves the primal SVM optimization problem, we just use the best response of the Optimizer player in terms of the optimal choice of $\vec{\lambda}$ for the Adversary, namely $\vec{w} = \sum_{i=1}^{n} \lambda_i y_i \vec{x}_i$.

Alas, there is no closed form solution to the dual SVM optimization problem (due to the non-negativity constraints on the Lagrange multipliers). However, there are efficient algorithms that can (approximately) solve the problem. Moreover, the form of the objective has a rather useful property, as we discuss next.

Kernel SVM

Notice that the objective of the dual SVM optimization problem only depends on the $\vec{x}_i$'s through their inner products with each other. This turns out to be very useful especially when combined with certain feature expansions $\vec{\phi}(\vec{x})$. For certain feature expansions, it is possible to compute inner products $\vec{\phi}(\vec{x}_i) \cdot \vec{\phi}(\vec{x}_j)$ more quickly than it is to compute the feature expansion $\vec{\phi}(\vec{x}_i)$. Therefore, it may be computationally beneficial to solve the dual SVM optimization problem for the Lagrange multipliers $\vec{\lambda}$, and implicitly represent a solution $\vec{w}$ to the primal SVM optimization problem using the form of the best response $\vec{w} = \sum_{i=1}^{n} \lambda_i y_i \vec{\phi}(\vec{x}_i)$ to $\vec{\lambda}$. This means you don't ever explicitly form the vector $\vec{w}$, but whenever you need to compute the inner product of $\vec{w}$ with (the feature expansion of) a new point $\vec{\phi}(\vec{x})$, you compute it using the formula

$$\vec{\phi}(\vec{x}) \cdot \vec{w} = \sum_{i=1}^{n} \lambda_i y_i (\vec{\phi}(\vec{x}) \cdot \vec{\phi}(\vec{x}_i)).$$

This formula never requires you to compute the feature expansion $\vec{\phi}(\vec{x})$ explicitly. Instead, you just have to compute inner products $\vec{\phi}(\vec{x}) \cdot \vec{\phi}(\vec{x}_i)$ for each $i = 1, \ldots, n$ (and do some simple arithmetic). This way of using of the solution to the dual SVM optimization problem is called Kernel SVM.

Complementary slackness

One more interesting aspect of the SVM optimization problem that comes out of this analysis of the dual problem is that the optimal weight vector $\vec{w}$, in terms of the optimal Lagrange multipliers $\vec{\lambda}$, only depends on the examples $(\vec{x}_i, y_i)$ for which $\lambda_i > 0$. This is clear from the form of the best response of the Optimizer:

$$\vec{w} = \sum_{i=1}^{n} \lambda_i y_i \vec{x}_i = \sum_{\substack{i \in \{1,\ldots,n\} \\ \text{s.t. } \lambda_i > 0}} \lambda_i y_i \vec{x}_i.$$

Which examples can have a positive Lagrange multiplier? Recall that in the version of the game where the Adversary gets to choose $\vec{\lambda}$ after seeing the choice of the Optimizer, the Adversary will always choose $\lambda_i = 0$ whenver $1 - y_i(\vec{x}_i \cdot \vec{w}) < 0$ (else the Optimizer would be better off and the Adversary would be worse off). So the only examples for which λ_i can be non-zero are those examples $(\vec{x}_i, y_i)$ for which $1 - y_i(\vec{x}_i \cdot \vec{w}) = 0$. Indeed, these are the feature vectors $\vec{x}_i$ closest to the decision boundary of the linear classifier determined by $\vec{w}$, and they are all at exactly the same distance from the decision boundary. These examples are called the "support vectors". It is these examples that determine the optimal weight vector $\vec{w}$. This can be particularly useful in the context of Kernel SVM, where one uses the implicit representation of $\vec{w}$ through the Lagrange multipliers and the training examples. Only the training examples that are support vectors are needed.

This is a special case of a property called complementary slackness; for the SVM optimization problem, it implies that the optimal choices of $\vec{w} \in \mathbb{R}^d$ and $\vec{\lambda} \geq \vec{0}$ satisfy

$$\lambda_i(1 - y_i(\vec{x}_i \cdot \vec{w})) = 0 \quad \text{for } i = 1, \ldots, n.$$