1 Sparsity in SVMs

When we solve for the solution to an SVM problem (constraints omitted for brevity),

$\min_{w, \xi} L(w, \xi) = ||w||^2 + C \sum_{i=1}^{n} \xi_i$

we observed that only those points near the decision boundary, the support vectors, contributed to the boundary. In other words, most slack variables $\xi_i$ were zero, and only those points which were support vectors had nonzero $\xi_i$’s. In other other words, the vector $\xi$ of slack variables is what we call sparse (most entries are 0). We are interested in explaining why this phenomenon occurs when solving this optimization problem.

To reason about this, you can think about what happens to the loss function with respect to the changing some slack variable $\frac{\partial L}{\partial \xi_i}$. A unit decrease in $\xi_i$ results in a “reward” of $C$. This is important because no matter what the current value of $\xi_i$ is, the reward for decreasing it is constant. On the other hand, the cost of decreasing it comes from the changes to $||w||^2$. At any point, the reward for decreasing $\xi_i$ is either going to be worth it (greater than cost incurred from $w$) or not worth it (less than cost incurred from $w$). So, intuitively, $\xi_i$ will continue to decrease until it hits zero (lower-bound) or, it will get stuck somewhere.

Compare this to the LS-SVM formulation (constraints omitted for brevity again),

$\min_{w, \xi} ||w||^2 + C \sum_{i=1}^{n} \xi_i^2$

Here, the reward for decreasing $\xi_i$ is no longer constant because the shape of the loss function with respect to $\xi_i$ is smooth rather than pointy. At any point, a unit decrease in $\xi_i$ results in a “reward” of $2C\xi_i$. That is, the reward changes with respect to the current value of $\xi_i$, and in particular, when $\xi_i$ approaches 0, the rewards get smaller and smaller: infinitesimal. On the other hand, the cost of decreasing slack, which comes from how $w$ changes, will be something finite. This tells us that before any $\xi_i$ hits zero, there will likely be a point when further decreasing it will no longer outweigh the cost for how $w$ changes, which is why the $\xi_i$’s will halt their descent before they hit zero.

In general, when we penalize one component of an optimization problem with non-squared loss of a pointy shape, this component will tend to be sparse.
2 LASSO

One reason that sparsity is desirable for the purpose of simplified evaluation of test points. Think about kernels. If we have reason to believe that the weight vector \( a \) is sparse, then after we compute \( a \) in training, we can throw away those training points with 0 weight, as they will contribute nothing to the evaluation of the hypothesized regression values of test points.

Similarly, if we were learning some weight vector \( w \) on features that we had reason to believe would be sparse, after we compute \( w \), we could throw away those features/dimensions of \( w \) entirely, allowing for faster evaluation of our hypothesis function on test points.

Let’s talk about a method of introducing sparsity to the weight vector in ordinary least squares. We saw from the SVM case that sparsity has something to do with the introduction of non-squared error penalties on the weight vector, so we will try something similar for the weight vector in OLS. The \textbf{LASSO} or \textbf{Linear Least Squares with L1 Regularization} is described by the following optimization problem:

\[
\min_w ||Xw - Y||^2 + \lambda ||w||_1
\]

where \( ||z||_1 \) is the L1-norm of \( z \), which is a sum of absolute values instead of squares.

\[
||z||_1 = \sum_{i=1}^{d} |z_i|
\]

You can imagine that for any particular component \( w_i \) of \( w \), the shape of the loss function is pointy. Specifically, the “reward” for decreasing \( w_i \) by a unit amount is a constant \( \lambda \). Thus, for the same argument as the slack variables in the SVM, this formulation will cause the resulting weight vector \( w \) to tend to be sparse.

This can be argued geometrically as well.

In both of these figures, we’re working with 2-dimensional data points and thus two components of the weight vector \( w \). In both of these figures, the red ellipses represent isocontours of the squared...
loss \( \|Xw - Y\|^2 \). On the left, there is a circle which represents an isocontour (for a particular value of \( \lambda \)) of \( \lambda \|w\|^2 \). This figure is supposed to reveal something about ridge regression. Specifically, we should realize that an optimal \( w \) will only occur at points of tangency between the red ellipse and the blue circle (otherwise we could move along the isocontour of one of the functions in some direction and improve the value of the objective function of the other function, thereby improving the overall value of the loss function). We can’t really say anything about these points of tangency other than the fact that the blue circle centered at the origin draws it closer to the origin (ridge regression penalizes large weights).

Now, look at the right hand side. The red ellipses represent the same thing, but now instead of a circle about the origin, we have a diamond. This is an isocontour of \( \lambda \|w\|_1 \). The same is true about optimal \( w \)’s: they need to occur at points of tangency between the ellipse and the diamond. The thing is: tangency is very likely to happen at the corners of the diamond because they are single points from which the rest of the diamond draws away from. And what are the corners of the diamond? Why, they are points at which one component of \( w \) is 0!