1 Support Vector Machines

So far, we’ve explored **generative models** (LDA) and **discriminative models** (logistic regression), but in both of these methods, we tasked ourselves with modeling some kind of probability distribution. One observation about classification is that in the end, if we only care about assigning each data point a class, all we really need to know where the decision boundaries are, and we can skip thinking about the distributions. **SVMs** are an attempt to model decision boundaries directly in this spirit.

Here’s the setup to the problem:

- You have training points $x_i \in \mathbb{R}^d$,
- Labels for them $y_i \in \{-1, +1\}$,
- And you are trying to find a hyperplane $H$ which separates the +1s from the -1s.

1.1 Hard-Margin SVM

In the case that your training data is **linearly separable**, meaning you can draw a $d - 1$-dimensional hyperplane that perfectly separates the +1s from the -1s, you can learn a **hard-margin** or **maximum-margin SVM**. Finding this hyperplane amounts to finding a weight vector $w$ such that for all $x_i$,

$$\begin{align*}
    w^T x_i &\geq 1 & \text{if } y_i = 1 \\
    w^T x_i &\leq -1 & \text{if } y_i = -1
\end{align*}$$

Equivalently, for all $x_i$

$$y_i (w^T x_i) \geq 1$$

If we can find such a $w$, then we can define our decision boundary $H$ to be $\{x : w^T x = 0\}$, the set of all points where $w^T x$ is zero. Thus if we were given a new data point $x$, we would classify $x$ as the sign of $w^T x$.

However, there are many such $w$. You can imagine that if you found one $w$ that satisfies the constraints, then $2w$ would also satisfy them as well. So, which one do we want?

The answer is: we want the one with the smallest Euclidean norm. Let’s talk about why. Look at this picture:
It should make intuitive sense that if we were to pick any line to separate the X’s from the C’s, a good choice would be the one that lies nearest to the center of the space which separates the two classes. Formally, this is the line which is equidistant to the closest points of different classes on either side of it. This distance is what we call the **margin**, and when we pick this line, what we are doing is **maximizing the margin**.

Maximizing the margin increases the generalization ability of the SVM. For example, in Figure ??, we consider another potential linear separator that is not max-margin. The closest point to this line is in class C, and one could imagine that if we observed new test points that are nearby, they should also be of class C. The pictured linear separator would incorrectly classify some of these new test points, while the max-margin separator would still be able to classify them.

Here’s an important fact that you should understand before reading on: **the margin of an SVM is determined by** \( w \). That is, only once you pick \( w \) do you know where and how big the margin of your SVM is. From the picture above, it’s easy to think that the margin is determined solely from the training points. This is not the case! Only the **best** margin is determined from the training points.

Okay, back to talking about hard-margin SVMs. How do we find the best hyperplane that maximizes the margin? Well, let’s try to write the expression for the size of the margin in terms of \( w \). From our constraints (the ones involving \( y_i \)), we know that the closest points of either class are at
least 1 contour line away from the $H$ as possible. That is to say, since we enforced $y_i(w^T x_i) \geq 1$, the closest a point $x_i$ can be to the hyperplane $w^T x = 0$ is on one of the hyperplanes $w^T x = \pm 1$. The distance between these two hyperplanes, then, is the size of our margin! With some linear algebra, we get that this distance is exactly $\frac{2}{||w||}$. And so, if we want to maximize it, we need only to minimize $||w||$, as previously stated.

Here’s an example of what minimizing $||w||$ accomplishes for us:

![Figure 3: Dotted: margin boundaries of $w = [2 2]$. Dashed: margin boundaries of $w = [1 1]$.](image)

Here, you can see that if we picked a $w$ with larger components (and thus, a larger Euclidean norm), then we only get to say that we have a margin that’s half the size as if we picked a $w$ with smaller components. This is because the level sets AKA contour lines for $w^T x$ are closer together when $||w||$ is larger.

Thus, our optimization problem for hard-margin SVMs is:

$$\text{Minimize } \frac{1}{2}||w||^2$$

subject to $y_i(w^T x_i) \geq 1$

### 1.2 Soft-margin SVMs

The hard-margin SVM optimization problem has a unique solution only if the data are linearly separable, but it has no solution otherwise. This is because the constraints are impossible to satisfy if we can’t draw a hyperplane that separates the +1s from the -1s. In addition, hard-margin SVMs are very sensitive to outliers - for example, if our data is class-conditionally distributed Gaussian such that the two Gaussians are far apart, if we witness an outlier from class +1 that crosses into the typical region for class -1, then hard-margin SVM will be forced to compromise a more generalizable fit in order to accommodate for this point. Thus, we have reason to study classifiers which don’t require their training points to be linearly separable and are robust to outliers. To this end, we’ll talk about what’s called the **soft-margin SVM**.
A soft-margin SVM modifies its constraints from the hard-margin SVM by allowing some points to violate the margin. Formally, it introduces what are called slack variables $\xi_i$, one for each training point, and sets its constraints as:

$$y_i(w^T x_i) \geq 1 - \xi_i$$

$$\xi_i \geq 0$$

which, you should understand, is a less-strict, softer version of the previous constraint because it says that point $x_i$ need only be a "distance" of $1 - \xi_i$ of the separating hyperplane instead of a hard "distance" of 1.

By the way, the Greek letter $\xi$ is spelled "xi" and pronounced "zai". $\xi_i$ is pronounced "zai-eye."

These constraints would be useless if we didn’t bound the values of the $\xi_i$’s, however, because by setting them to large values, we are essentially saying that any point may violate the margin by an arbitrarily large distance... which makes our choice of $w$ meaningless. Thus, we modify the objective function to be:

$$\text{Minimize } \frac{1}{2}||w||^2 + C\sum_{i=1}^n \xi_i$$

Where $C$ is a hyperparameter meant to be tuned through cross-validation. Below is a nice table that summarizes the properties of having large $C$ versus small $C$. It should make sense that as $C$ goes to infinity, the penalty for having non-zero $\xi_i$ goes to infinity, and thus we force the $\xi_i$’s to be zero, which is exactly the setting of the hard-margin SVM.

<table>
<thead>
<tr>
<th></th>
<th>small $C$</th>
<th>large $C$</th>
</tr>
</thead>
<tbody>
<tr>
<td>desire</td>
<td>maximize margin</td>
<td>keep $\xi_i$’s small or zero</td>
</tr>
<tr>
<td>danger</td>
<td>underfitting</td>
<td>overfitting</td>
</tr>
<tr>
<td>outliers</td>
<td>less sensitive</td>
<td>more sensitive</td>
</tr>
</tbody>
</table>

1.3 SVMs as Tikhonov Regularization Learning

Consider the following regularized regression problem:

$$\min \frac{1}{n} \sum_{i=1}^n L(y_i, w^T x_i) + \lambda ||w||^2$$

In the context of classification, the loss function that we would like to optimize is 0-1 (step) loss:

$$L(y, w^T x) = \begin{cases} 1 & y(w^T x) < 0 \\ 0 & y(w^T x) \geq 0 \end{cases}$$

The 0-1 loss is 0 if $x$ is correctly classified and 1 otherwise. Thus minimizing $\frac{1}{n} \sum_{i=1}^n L(y_i, w^T x_i)$ is directly minimizing classification error on the training set. However, the 0-1 loss is difficult to optimize: it is neither convex nor differentiable (see Figure ??).
Figure 4: Step (0-1) loss, hinge loss, and squared loss. Squared loss is convex and differentiable, hinge loss is only convex, and step loss is neither.

One can try to modify the 0-1 loss to be convex. The points with \( y(w^T x) \geq 0 \) should remain at 0 loss, but we may consider allowing for different values of penalty for misclassified points. This leads us to the hinge loss, as illustrated in Figure ??:

\[
L_{hinge}(y, w^T x) = \max(1 - y(w^T x), 0)
\]

Thus the regularized regression problem becomes

\[
\min \frac{1}{n} \sum_{i=1}^{n} \max(1 - y_i(w^T x_i), 0) + \lambda \|w\|^2
\]

Recall that the original soft-margin SVM optimization problem is

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

\[
y_i(w^T x_i) \geq 1 - \xi_i
\]

\[
\xi_i \geq 0
\]

We claim these two formulations are actually equivalent. By manipulating the first constraint, we get the constraint

\[
\xi_i \geq 1 - y_i(w^T x_i)
\]

Combining with the constraint \( \xi_i \geq 0 \), we get the constraint

\[
\xi_i \geq \max(1 - y_i(w^T x_i), 0)
\]

At the optimal value of the optimization problem, these inequalities must be tight. Otherwise, we could lower each \( \xi_i \) to equal \( \max(1 - y_i(w^T x_i), 0) \) and decrease the value of the objective function. Thus we can rewrite the soft-margin SVM optimization problem as

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

\[
\xi_i = \max(1 - y_i(w^T x_i), 0)
\]

If we divide by \( Cn \) (which does not change the optimal solution of the optimization problem), we can see that this formulation is equivalent to the regularized regression problem, with \( \lambda = \frac{1}{2cn} \).
Thus we have two interpretations of soft-margin SVM: either as finding a max-margin hyperplane that is allowed to make some mistakes via slack variables $\xi_i$, or as regularized empirical risk minimization with the hinge loss.