Week 7: Multiclass Support Vector Machines

Instructor: Sergey Levine

1 Support vector machines recap

The support vector machine (SVM) optimization is defined as

$$\begin{aligned}
&\min_{\mathbf{w},w_0,s_1,\dots,s_N} \frac{1}{2}||\mathbf{w}||^2 + \lambda \sum_{i=1}^{N} s_i \text{ such that} \\
&y^i(\mathbf{w}\cdot h(\mathbf{x}^i) + w_0) + s_i \geq 1 \quad \forall i \in \{1,\dots,N\} \\
&s_i \geq 0 \quad \forall i \in \{1,\dots,N\}
\end{aligned}$$

As we saw in the previous lecture, solving this optimization recovers a linear classifier of the form $y = \text{sign}(\mathbf{w}\cdot h(\mathbf{x}) + w_0)$ that minimizes the hinge loss for all misclassified points and maximizes the size of the margin (the distance to the closest point to the decision boundary). The term "support vector" refers to the vectors from the decision boundary to the closest points. Note that moving any point that is correct classified and further from the decision boundary than the margin will not affect the optimal weights, hence the term "support vector:" these vectors "support" the boundary, while all others do not.

2 Multiclass SVMs

Lastly, we'll briefly discuss how we can use SVMs when we have more than two classes. There are two main approaches we'll discuss: (1) one-against-all classifiers and (2) multiclass SVMs.

One-against-all classification is the simplest way to adapt SVMs to multiclass classification. In this scheme, instead of solving a single learning problem with L_y classes, we instead solve L_y binary problems, each of which requires us to classify the current class j against *all* other classes. So we simply construct L_y datasets, for each of which the label is $y^i_j = \delta(y^i = j)$, and we get L_y weight vectors $\mathbf{w}_1, \dots, \mathbf{w}_{L_y}$. Now we just need to figure out how to classify a new point $\mathbf{x}^\star$. The idea is very simple: the further the point is from the decision boundary in the "positive" direction, the more likely we think it is to belong to that class. So we simply choose the point for which the point is furthest from the boundary in the positive direction, and set the class according to:

$$y^\star = \arg\max_j h(\mathbf{x}^\star)\cdot \mathbf{w}_j + w_{0,j}.$$

One-against-all classification is reasonable and can work quite well, though it requires training multiple SVMs. We can also train a single SVM to perform multiclass classification directly, though this is a little bit more complex.

The intuition behind the multiclass SVM is that, if our classification rule is

$$y^i = \arg\max_j h(\mathbf{x}^i) \cdot \mathbf{w}_j + w_{0,j},$$

we should simply make sure that if $y^i = j$, then $h(\mathbf{x}^i) \cdot \mathbf{w}_j + w_{0,j}$ is greater than $h(\mathbf{x}^i) \cdot \mathbf{w}_{j'} + w_{0,j'}$ for all $j' \neq j$ by the largest margin, in the same way that we make sure that $h(\mathbf{x}^i) \cdot \mathbf{w}_j + w_{0,j} \geq 1$ in the binary SVM. So we can directly optimize over all of our decision boundaries with constraints that enforce it, and the same objective as before (but now summed over all decision boundaries):

$$\min_{\mathbf{w}_1,\ldots,\mathbf{w}_{L_1},w_{0,1},\ldots,w_{0,L_y},s_1,\ldots,s_N} \frac{1}{2}\sum_{j=1}^{L_y} ||\mathbf{w}_j||^2 + \lambda \sum_{i=1}^{N} s_i \text{ such that}$$

$$\mathbf{w}_{y^i} \cdot h(\mathbf{x}^i) + w_{0,y^i} + s_i \geq \mathbf{w}_{j'} \cdot h(\mathbf{x}^i) + w_{0,j'} + 1 \quad \forall i \in \{1,\ldots,N\}, \forall j' \neq y^i$$

$$s_i \geq 0 \quad \forall i \in \{1,\ldots,N\}$$