Perceptron, convergence, and generalization

Recall that we are dealing with linear classifiers through origin, i.e.,

$$f(\mathbf{x};\theta) = \text{sign}\big(\,\theta^T\mathbf{x}\,\big) \tag{1}$$

where $\theta \in \mathcal{R}^d$ specifies the parameters that we have to estimate on the basis of training examples (images) $\mathbf{x}_1, \ldots, \mathbf{x}_n$ and labels $y_1, \ldots, y_n$.

We will use the perceptron algorithm to solve the estimation task. Let k denote the number of parameter updates we have performed and $\theta^{(k)}$ the parameter vector after k updates. Initially $k = 0$ and $\theta^{(k)} = 0$. The algorithm then cycles through all the training instances $(\mathbf{x}_t, y_t)$ and updates the parameters only in response to mistakes, i.e., when the label is predicted incorrectly. More precisely, we set $\theta^{(k+1)} = \theta^{(k)} + y_t\mathbf{x}_t$ when $y_t(\theta^{(k)})^T\mathbf{x}_t < 0$ (mistake), and otherwise leave the parameters unchanged.

Convergence in a finite number of updates

Let's now show that the perceptron algorithm indeed convergences in a finite number of updates. The same analysis will also help us understand how the linear classifier generalizes to unseen images. To this end, we will assume that all the (training) images have bounded Euclidean norms, i.e., $\|\mathbf{x}_t\| \leq R$ for all t and some finite R. This is clearly the case for any pixel images with bounded intensity values. We also make a much stronger assumption that there exists a linear classifier in our class with finite parameter values that correctly classifies all the (training) images. More precisely, we assume that there is some $\gamma > 0$ such that $y_t(\theta^*)^T\mathbf{x}_t \geq \gamma$ for all $t = 1, \ldots, n$. The additional number $\gamma > 0$ is used to ensure that each example is classified correctly with a *finite margin*.

The convergence proof is based on combining two results: 1) we will show that the inner product $(\theta^*)^T\theta^{(k)}$ increases at least linearly with each update, and 2) the squared norm $\|\theta^{(k)}\|^2$ increases at most linearly in the number of updates k. By combining the two we can show that the cosine of the angle between $\theta^{(k)}$ and θ^* has to increase by a finite increment due to each update. Since cosine is bounded by one, it follows that we can only make a finite number of updates.

Part 1: we simply take the inner product $(\theta^*)^T\theta^{(k)}$ before and after each update. When making the k^{th} update, say due to a mistake on image $\mathbf{x}_t$, we get

$$(\theta^*)^T\theta^{(k)} = (\theta^*)^T\theta^{(k-1)} + y_t(\theta^*)^T\mathbf{x}_t \geq (\theta^*)^T\theta^{(k-1)} + \gamma \tag{2}$$

since, by assumption, $y_t(\theta^*)^T\mathbf{x}_t \geq \gamma$ for all t (θ^* is always correct). Thus, after k updates,

$$(\theta^*)^T\theta^{(k)} \geq k\gamma \tag{3}$$

Part 2: Our second claim follows simply from the fact that updates are made only on mistakes:

$$\begin{aligned}
\|\theta^{(k)}\|^2 &= \|\theta^{(k-1)} + y_t\mathbf{x}_t\|^2 && (4)\\
&= \|\theta^{(k-1)}\|^2 + 2y_t(\theta^{(k-1)})^T\mathbf{x}_t + \|\mathbf{x}_t\|^2 && (5)\\
&\leq \|\theta^{(k-1)}\|^2 + \|\mathbf{x}_t\|^2 && (6)\\
&\leq \|\theta^{(k-1)}\|^2 + R^2 && (7)
\end{aligned}$$

since $y_t(\theta^{(k-1)})^T\mathbf{x}^t < 0$ whenever an update is made and, by assumption, $\|\mathbf{x}_t\| \leq R$. Thus,

$$\|\theta^{(k)}\|^2 \leq kR^2 \tag{8}$$

We can now combine parts 1) and 2) to bound the cosine of the angle between θ^* and $\theta^{(k)}$:

$$cos(\theta^*, \theta^{(k)}) = \frac{(\theta^*)^T\theta^{(k)}}{\|\theta^{(k)}\|\,\|\theta^*\|} \overset{1)}{\geq} \frac{k\gamma}{\|\theta^{(k)}\|\,\|\theta^*\|} \overset{2)}{\geq} \frac{k\gamma}{\sqrt{kR^2}\,\|\theta^*\|} \tag{9}$$

Since cosine is bounded by one, we get

$$1 \geq \frac{k\gamma}{\sqrt{kR^2}\,\|\theta^*\|} \text{ or } k \leq \frac{R^2\|\theta^*\|^2}{\gamma^2} \tag{10}$$

Margin and geometry

It is worthwhile to understand this result a bit further. For example, does $\|\theta^*\|^2/\gamma^2$ relate to how difficult the classification problem is? Indeed, it does. We claim that its inverse, i.e., $\gamma/\|\theta^*\|$ is the smallest distance in the image space from any example (image) to the decision boundary specified by θ^*. In other words, it serves as a measure of how well the two classes of images are separated (by a linear boundary). We will call this the geometric margin or γ_{geom} (see figure 1). γ_{geom}^{-1} is then a fair measure of how difficult the problem is: the smaller the geometric margin that separates the training images, the more difficult the problem.

To calculate γ_{geom} we measure the distance from the decision boundary ${\theta^*}^T\mathbf{x} = 0$ to one of the images $\mathbf{x}_t$ for which $y_t{\theta^*}^T\mathbf{x}_t = \gamma$. Since θ^* specifies the normal to the decision boundary,

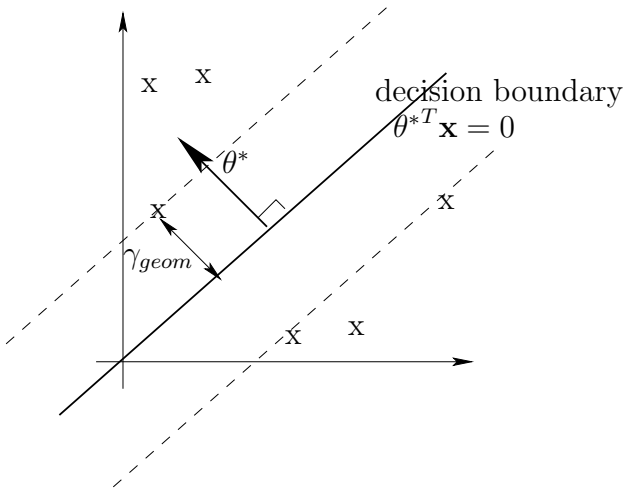

Figure 1: Geometric margin

the shortest path from the boundary to the image $\mathbf{x}_t$ will be parallel to the normal. The image for which $y_t \theta^{*T} \mathbf{x}_t = \gamma$ is therefore among those closest to the boundary. Now, let's define a line segment from $\mathbf{x}(0) = \mathbf{x}_t$, parallel to θ^*, towards the boundary. This is given by

$$\mathbf{x}(\xi) = \mathbf{x}(0) - \xi \frac{y_t \theta^*}{\|\theta^*\|} \tag{11}$$

where ξ defines the length of the line segment since it multiplies a unit length vector. It remains to find the value of ξ such that $\theta^{*T}\mathbf{x}(\xi) = 0$, or, equivalently, $y_t \theta^{*T}\mathbf{x}(\xi) = 0$. This is the point where the segment hits the decision boundary. Thus

$$\begin{aligned} y_t \theta^{*T}\mathbf{x}(\xi) &= y_t \theta^{*T}\left[\mathbf{x}(0) - \xi \frac{y_t \theta^*}{\|\theta^*\|}\right] && (12)\\ &= y_t \theta^{*T}\left[\mathbf{x}_t - \xi \frac{y_t \theta^*}{\|\theta^*\|}\right] && (13)\\ &= y_t \theta^{*T}\mathbf{x}_t - \xi \frac{\|\theta^*\|^2}{\|\theta^*\|} && (14)\\ &= \gamma - \xi\|\theta^*\| = 0 && (15) \end{aligned}$$

implying that the distance is exactly $\xi = \gamma/\|\theta^*\|$ as claimed. As a result, the bound on the number of perceptron updates can be written more succinctly in terms of the geometric

margin γ_{geom} (distance to the boundary):

$$k \leq \left(\frac{R}{\gamma_{geom}} \right)^2 \tag{16}$$

with the understanding that γ_{geom} is the largest geometric margin that could be achieved by a linear classifier for this problem. Note that the result does not depend (directly) on the dimension d of the examples, nor the number of training examples n. It is nevertheless tempting to interpret $\left(\frac{R}{\gamma_{geom}} \right)^2$ as a measure of difficulty (or complexity) of the problem of learning linear classifiers in this setting. You will see later in the course that this is exactly the case, cast in terms of a measure known as *VC-dimension*.

Generalization guarantees

We have so far discussed the perceptron algorithm only in relation to the training set but we are more interested in how well the perceptron classifies images we have not yet seen, i.e., how well it generalizes to new images. Our simple analysis above actually provides some information about generalization. Let's assume then that all the images and labels we could possibly encounter satisfy the same two assumptions. In other words, 1) $\|\mathbf{x}_t\| \leq R$ and 2) $y_t \theta^{*T} \mathbf{x}_t \geq \gamma$ for all t and some finite θ^*. So, in essence, we assume that there is a linear classifier that works for all images and labels in this problem, we just don't know what this linear classifier is to start with. Let's now imagine getting the images and labels one by one and performing only a single update per image, if misclassified, and move on. The previous situation concerning the training set corresponds to encountering the same set of images repeatedly. How many mistakes are we now going to make in this infinite arbitrary sequence of images and labels, subject only to the two assumptions? The same number $k \leq (R/\gamma_{geom})^2$. Once we have made this many mistakes we would classify all the new images correctly. So, provided that the two assumptions hold, especially the second one, we obtain a nice guarantee of generalization. One caveat here is that the perceptron algorithm does need to know when it has made a mistake. The bound is after all cast in terms of the number of updates based on mistakes.

Maximum margin classifier?

We have so far used a simple on-line algorithm, the perceptron algorithm, to estimate a linear classifier. Our reference assumption has been, however, that there exists a linear classifier that has a large geometric margin, i.e., whose decision boundary is well separated

from all the training images (examples). Can't we find such a large margin classifier directly? Yes, we can. The classifier is known as the Support Vector Machine or SVM for short. See the next lecture for details.