CS281B/Stat241B (Spring 2008) Statistical Learning Theory **Lecture: 3**

Minimax risk bounds for linear threshold functions

Lecturer: Peter Bartlett *Scribe: Hao Zhang*

1 Review

We assume that there is a probability distribution P on $\mathcal{X} \times \mathcal{Y}$, and that the pairs $(X_1, Y_1), \ldots, (X_n, Y_n)$ and (X, Y) are chosen independently according to P. The aim is to choose f so that the *risk* of f,

$$R(f) = \mathbb{E}\ell(f(X), Y),$$

is small. For instance, in the pattern classification example, $\mathcal{Y} = \{\pm 1\}$, and the risk is the misclassification probability.

$$R(f) = \mathbb{E}1[f(X) \neq Y] = \Pr(f(X) \neq Y).$$

One family of approaches to pattern classification problems is to fix a class F of functions that map from $\mathcal{X}$ to $\mathcal{Y}$ and choose f_n from F. For example, consider the class of linear threshold functions on $\mathcal{X} = \mathbb{R}^d$,

$$F = \left\{ x \mapsto \mathrm{sign}(\theta' x) : \theta \in \mathbb{R}^d \right\}.$$

The decision boundaries are hyperplanes through the origin ($d-1$-dimensional subspaces), and the decision regions are half-spaces.

We have seen that, when there is a linear threshold function that classifies all of the training data correctly, the perceptron algorithm terminates after a finite number of mistakes and returns such a function. The number of mistakes depends on the scale of the margin between the two classes.

Although the perceptron algorithm performs well when the margin is large, notice that it does not necessarily maximize the margin. (Later, we'll consider other methods that do maximize the margin.)

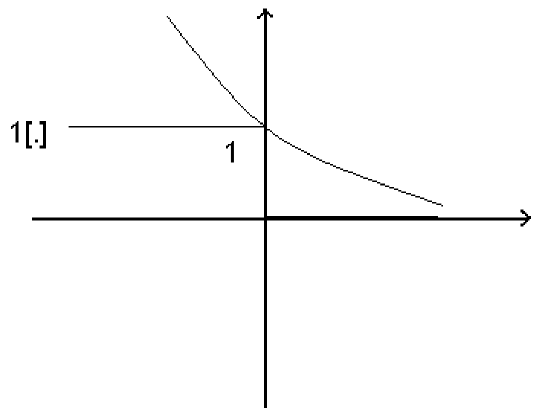
The perceptron algorithm converges only when there is a linear threshold function that correctly classifies all of the training data. In those cases, it can be viewed as choosing a linear function $g(x) = \theta' x$ such that the risk of the decision function, $f(x) = \mathrm{sign}(g(x))$, is zero, and hence minimal. A question was asked about what can be done if there is no perfect linear threshold function. Notice that we can write

$$R(f(X)) = R(\mathrm{sign}(g(X))) \leq Pr(Yg(X) \leq 0) = \mathbb{E}(1[Yg(X) \leq 0]).$$

If G denotes the set of linear functions, then the perceptron algorithm can be thought of as minimizing over G the sample average of the indicator function of a misclassification, that is,

$$\min_{g \in G} \hat{\mathbb{E}}(1[Yg(X) \leq 0]),$$

where $\hat{\mathbb{E}}$ denotes the expectation under the empirical distribution. In general, if the minimum of this expectation is non-zero, this minimization problem is intractable. An alternative is to replace the indicator function $1[\cdot \leq 0]$ with some convex loss function $\phi(\cdot)$. This loss function should penalize negative values of its argument so as to favor solutions for which Y and $g(X)$ tend to have the same sign. Typically, ϕ is some kind of convex approximation of the step function, as indicated in the figure below. This is the approach taken by the SVM (with ϕ the hinge loss) and AdaBoost (with ϕ the exponential loss).



2 A Minimax lower bound on risk

We made the observation that if $n \le d$, then we should anticipate difficulties. In particular, if $x_1, \ldots, x_n$ are linearly independent, then for any $y_1, \ldots, y_n$, we can find $\theta \in \mathbb{R}^d$ such that

$$\theta' \left[x_1 | x_2 | \cdots | x_n\right] = [y_1, y_2, \ldots, y_n],$$

and hence $\mathrm{sign}(\theta' x_i) = y_i$. If we can fit *any* labels, we should not expect to predict the labels of subsequent points accurately.

The following theorem makes this precise, by showing that, under these conditions, any method will perform poorly.

Theorem 2.1. For any $n \ge 1$ and any mapping $f_n : \mathbb{R}^d \times \left(\mathbb{R}^d \times \{\pm 1\}\right)^n \to \{\pm 1\}$, there is a probability distribution on $\mathbb{R}^d \times \{\pm 1\}$ for which some linear threshold function $f \in F$ has $R(f) = 0$ but

$$\mathbb{E}R(f_n) \ge \min\left(\frac{n-1}{2n}, \frac{d-1}{2n}\right)\left(1 - \frac{1}{n}\right)^n.$$

This is an example of a minimax lower bound, since it gives a lower bound on

$$\min_{f_n} \max_{P} \mathbb{E}R(f_n),$$

where the max is over all P for which some $f \in F$ has zero risk, and the min is over all methods f_n—not just the perceptron algorithm, or other algorithms that predict with linear threshold functions, but any deterministic prediction rule.

One thing to notice about the result is that the probability distribution is allowed to depend on n. So it's not saying that, for some distribution, the risk must decrease to zero at slower than a $1/n$ rate.

Proof. The broad idea of the proof is to use the probabilistic method: choose the probability distribution P uniformly at random from a class $\mathcal{P}$, and show that the expectation of $R(f_n)$ (expectation both under this random choice and under the choice of the data) is large. This implies that there is a distribution in the class that makes $R(f_n)$ large. Notice that this kind of proof is not constructive: we don't find out which distribution is bad for a particular algorithm f_n (like the perceptron algorithm). Certainly the bad distribution must depend on f_n. Randomization is an elegant way of showing that each f_n must fail in some case.

Each distribution in the class satisfies two properties:

1. There is a linear threshold function f with $R(f) = 0$. To achieve this, we fix a linearly independent set $S = \{z_1, \dots, z_n\} \in \mathbb{R}^d$. Then we restrict our marginal distributions on $\mathcal{X}$ to have support in S. By the observation above, for any $b = (b_1, \dots, b_d) \in \{\pm\}^d$, we can find a linear threshold function f_b satisfying $f_b(z_i) = b_i$ for all i.
2. A sample of size n is unlikely to contain much information about f. To do this, we'll concentrate most of the probability on a single point, say z_d, and make the others unlikely.

For each $b \in \{\pm 1\}^d$, define the probability distribution P_b via

$$P_b(x, y) = \begin{cases} \frac{\epsilon}{d-1} & \text{if } (x, y) = (z_i, b_i) \text{ for } i = 1, \dots, d-1, \\ 1 - \epsilon & \text{if } (x, y) = (z_d, b_d), \\ 0 & \text{otherwise.} \end{cases}$$

We'll choose b uniformly at random. Under this choice of b and of the data, we have

$$\mathbb{E}R(f_n) = \sum_{k=0}^{d-1} \mathbb{E}\left[R(f_n) | |U| = k\right] \Pr(|U| = k),$$

where $U = \{z_1, \dots, z_{d-1}\} - \{X_1, \dots, X_n\}$ is the set of 'light' elements of S that are unseen. The key observation is that, for these unseen elements, the corresponding b_i might as well have been chosen afterwards, since they are independent of the earlier choices. So on those points, the decision rule can do no better than tossing a coin. Formally, we can write

$$\mathbb{E}\left[R(f_n) | |U| = k\right] = \mathbb{E}\left[\mathbb{E}\left[\ell(f_n(X), Y) | \text{data, } |U| = k\right] | |U| = k\right],$$

and

$$\begin{aligned} \mathbb{E}\left[\ell(f_n(X), Y) | \text{data}\right] &= \sum_{i=1}^{d} \mathbb{E}\left[\ell(f_n(z_i), b_i) | \text{data}\right] \Pr(X = z_i) \\ &\geq \sum_{z_i \in U} \mathbb{E}\left[\ell(f_n(z_i), b_i) | \text{data}\right] \Pr(X = z_i) \\ &= |U| \times \frac{1}{2} \times \frac{\epsilon}{d-1}. \end{aligned}$$

Combining, we have

$$\mathbb{E}R(f_n) \geq \sum_{k=0}^{d-1} \frac{k\epsilon}{2(d-1)} \Pr(|U| = k), = \frac{\epsilon}{2(d-1)} \mathbb{E}|U|.$$

But the expected number of unseen light elements is

$$\begin{aligned} \mathbb{E}|U| &= \mathbb{E} \sum_{i=1}^{d-1} 1\left[z_i \notin \{X_1, \dots, X_n\}\right] \\ &= \sum_{i=1}^{d-1} \Pr\left(z_i \notin \{X_1, \dots, X_n\}\right) \\ &= (d-1)\left(1 - \frac{\epsilon}{d-1}\right)^n. \end{aligned}$$

Thus,

$$\mathbb{E}R(f_n) \geq \frac{\epsilon}{2}\left(1 - \frac{\epsilon}{d-1}\right)^n.$$

And now we just need to choose appropriate values of ϵ. If $n \geq d-1$, choose $\epsilon = (d-1)/n$ shows that $\mathbb{E}R(f_n) \geq (d-1)(1-1/n)^n/(2n)$. If $n < d-1$, choose $\epsilon = (n-1)/n (< (d-1)/n)$ shows that

$$\mathbb{E}R(f_n) \geq \frac{n-1}{2n}\left(1 - \frac{n-1}{(d-1)n}\right)^n \geq \frac{n-1}{2n}\left(1 - \frac{1}{n}\right)^n.$$

□

We'll see that, if n is large compared to d, then over the set of linear threshold functions, the empirical risk is uniformly close to the true risk, and so any linear function with zero empirical risk must have small risk. In particular, if the perceptron algorithm terminates, the function that it returns will have small risk.

Is d the right measure of the complexity of the class of linear threshold functions? Not always. If there is a large margin classifier, so that the perceptron converges quickly, then the solution it finds incorporates only few (r^2/δ^2) of the n (x_i, y_i) pairs. If this number is small compared to n, then the data has been *compressed* in some sense: the algorithm could have seen just this subset and produced the same (empirically correct) classifier.

The following theorem shows that the risk of the classifier returned by (something like) the perceptron algorithm is small whenever n is large compared to (something like) r^2/δ^2.

Theorem 2.2. Suppose that, for some $r > 0$, $\theta \in \mathbb{R}^d$ and $\delta > 0$, we have, almost surely,

$$\|X\| \leq r \qquad \text{and} \qquad \frac{Y\theta' X}{\|\theta\|} \geq \delta.$$

Consider the following randomized variant of the perceptron algorithm: Choose M uniformly from $\{1, \ldots, n\}$. Pass the initial data subsequence $(X_1, Y_1), \ldots, (X_M, Y_M)$ to the perceptron algorithm. Let f_n be the linear threshold function that it returns. Then

$$\mathbb{E}R(f_n) \leq \frac{r^2}{n\delta^2}.$$

(Notice that the expectation includes the randomization of the algorithm.)

Proof. Explicitly writing the expectation over the random choice of M shows that

$$\begin{aligned}\mathbb{E}R(f_n) &= \frac{1}{n}\sum_{m=1}^{n} \mathbb{E}\ell(f_m(X; X_1, Y_1, \ldots, X_m, Y_m), Y) \\ &= \frac{1}{n}\sum_{m=1}^{n} \mathbb{E}\ell(f_m(X_{m+1}; X_1, Y_1, \ldots, X_m, Y_m), Y_{m+1}) \\ &= \mathbb{E}\frac{1}{n}\sum_{m=1}^{n} \ell(f_m(X_{m+1}; X_1, Y_1, \ldots, X_m, Y_m), Y_{m+1}),\end{aligned}$$

where the second equality follows from the fact that (X_i, Y_i) and (X, Y) have the same distribution. (Let (X_{m+1}, Y_{m+1}) denote (X, Y).) Now consider

$$\frac{1}{n}\sum_{m=1}^{n} \ell(f_m(X_{m+1}; X_1, Y_1, \ldots, X_m, Y_m), Y_{m+1}).$$

With probability 1, the data will all satisfy the constraint on the radius and the margin. The perceptron convergence theorem shows that, with this data, the total number of iterations of the algorithm cannot exceed r^2/δ^2. Consider the perceptron algorithm when the misclassified pair chosen for the update is the one with smallest index. Then the total number of updates, U, satisfies

$$\sum_{m=1}^{n} \ell(f_m(X_{m+1}; X_1, Y_1, \ldots, X_m, Y_m), Y_{m+1}) \leq U,$$

because this sum ignores the updates that are made on a pair with an index that is smaller than the biggest update index so far. And since $U \leq r^2/\delta^2$, we have $\mathbb{E}R(f_n) \leq r^2/(n\delta^2)$. $\square$

And we can extend the minimax lower bound to show that, in a minimax sense, this is the best that we can hope for (up to constants) for a probability distribution satisfying the radius and margin conditions. We'll see this in the next lecture.