6

Probably Approximately Correct

As before, we consider a fixed *dictionary* $\mathcal{H}$ and select one classifier $\hat{h}_n$ that optimizes the empirical risk $\hat{R}(h)$. Recall:

- The *empirical risk* $\hat{R}$ and the classifier $\hat{h}_n$ depend on the *data* (X_i, Y_i), $1 \leq i \leq n$, and are random variables. In particular, they depend on n;
- The *risk* $R(h) = \mathbb{E}[\hat{R}(h)]$ depends on the underlying distribution on $\mathcal{X} \times \mathcal{Y}$, but not on n.

We have seen that if $\mathcal{H} = \{h_1, \ldots, h_K\}$ is finite, then with probability $1 - \delta$, we have

$$R(\hat{h}_n) - \inf_{h \in \mathcal{H}} R(h) \leq \sqrt{\frac{2\log(K) + 2\log(2/\delta)}{n}}. \tag{6.1}$$

Note that $\log(K)$ is proportional to the *bit size* of K: this is the amount of bits needed to represent numbers up to K, and can be seen as a measure of complexity for the set $\mathcal{H}$ (the "space" necessary to represent K elements). Bounds such as (6.1) are called **generalization bounds**.

Probably Approximately Correct Learning

An alternative point of view to generalization bounds would be to ask, for given **accuracy** $\epsilon > 0$ and **confidence** $\delta \in (0, 1)$, how many samples n are needed to get an accuracy of ϵ with confidence δ:

$$\mathbb{P}(R(\hat{h}_n) - \inf_{h \in \mathcal{H}} R(h) \leq \epsilon) \geq 1 - \delta.$$

Assuming $h^* \in \mathcal{H}$ and $Y = f(X)$, we have $R(h^*) = 0$, and h^* would be the *correct* classifier. The classifier $\hat{h}_n$ is then **probably** (with probability $1 - \delta$) **approximately** (up to an misclassification probability of at most ϵ) **correct**. This leads us to the notion of **Probably Approximately Correct** (**PAC**) learning. In what follows, we denote

by $\text{size}(\mathcal{H})$ the complexity of representing an element of $\mathcal{H}$. This is not a precise definition, but depends on the case at hand. For example, if $\mathcal{H} = \{h_1, \ldots, h_K\}$ is a finite set, then we can index this set using K numbers. On a computer, numbers up to K can be represented as binary numbers using $\lceil \log_2(K) \rceil$ bits, and hence (up to a constant factor) $\text{size}(\mathcal{H}) = \log(K)$ would be adequate here. Similarly, we denote by $\text{size}(\mathcal{X})$ the complexity of representing an element of the input space. For example, if $\mathcal{X} \subset \mathbb{R}^d$, then we would use d as size parameter (possibly multiplied by a constant factor to account for the size of representing a real number in floating point arithmetic on a computer). Note that $\text{size}(\mathcal{X})$ or $\text{size}(\mathcal{H})$ is not the same as the cardinality of these sets!

Definition 6.1. (PAC Learning [1]) A hypothesis class $\mathcal{H}$ is called **PAC-learnable** if there exists a classifier $\hat{h}_n \in \mathcal{H}$ depending on n random samples (X_i, Y_i), $i \in \{1, \ldots, n\}$, and a polynomial function $p(x, y, z, w)$, such that for any $\epsilon > 0$ and $\delta \in (0, 1)$, for *all* distributions on $\mathcal{X} \times \mathcal{Y}$,

$$\mathbb{P}\left(R(\hat{h}_n) \leq \inf_{h \in \mathcal{H}} R(h) + \epsilon \right) \geq 1 - \delta$$

holds whenever $n \geq p(1/\epsilon, 1/\delta, \text{size}(\mathcal{X}), \text{size}(\mathcal{H}))$. We also say that $\mathcal{H}$ is **efficiently** PAC-learnable, if the algorithm that produces $\hat{h}_n$ from the data runs in time polynomial in $1/\epsilon$, $1/\delta$, $\text{size}(\mathcal{X})$ and $\text{size}(\mathcal{H})$.

Remark 6.2. In our context, to say that an algorithm "runs in time $p(n)$" means that the number of steps, with respect to some suitable model of computation, is bounded by $p(n)$. Note that in this definition we disregard specific constants in the lower bound on n, but only require that it is polynomial. In computer science, polynomial time or space is considered *efficient*, while problems that require exponential time and/or space to solve are considered inefficient. For example, sorting n numbers can be performed in $O(n \log(n))$ operations and is efficient, while it is not known if finding the shortest route through n cities (the Traveling Salesman Problem) can be solved in a number of computational steps that is polynomial in n. This is the subject of the famous P vs NP conjecture.

In the case of a finite hypothesis space $\mathcal{H}$ with K elements, we have seen that $\mathcal{H}$ is PAC-learnable, since

$$n \geq \left(\frac{2}{\epsilon^2} \right) \left(\log(K) + \log\left(\frac{2}{\delta} \right) \right),$$

which is polynomial in all the relevant parameters. Under Massart's noise condition (see Chapter 5), we see that

$$n \geq \frac{1}{\gamma\epsilon} \left(\log(K) + \log\left(\frac{1}{\delta} \right) \right),$$

[1] In some references, such as the book "Foundations of Machine Learning" by Mohri, Rostamizadeh and Talwalkar, this version of PAC learning is called *Agnostic PAC Learning.*

samples are necessary to approximate the Bayes classifier up to a factor of ϵ with confidence $1 - \delta$. We also see here that the number of samples needed increases as $\gamma \to 0$, reflecting the fact that in the presence of high uncertainty, more observations are needed than if we have low uncertainty.

Learning Rectangles

So far we have considered learning with finite dictionaries of classifiers $\mathcal{H}$. We now illustrate an example where $\mathcal{H}$ is not finite, and show how PAC-learnability and generalization bounds can be derived in this setting.

Assume that our data describes a rectangle: the input space $\mathcal{X}$ is a subset of $\mathbb{R}^2$, and the function $f\colon \mathcal{X} \to \{0, 1\}$ is the indicator function of a closed rectangle B, so that for any point x, $f(x) = 1$ if x is in the rectangle, and 0 if not. Suppose that all we have access to is a random sample of labelled points, (x_i, y_i), $i \in \{1, \ldots, n\}$ (see Figure 6.1, left panel).

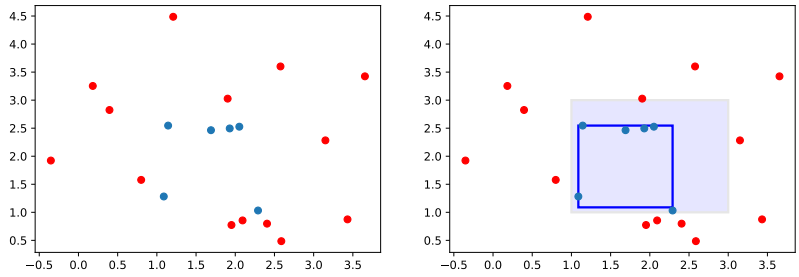

Figure 6.1: The blue points are in the true rectangle, while the red points are outside of it. The smaller blue rectangle represents the computed classifier $\hat{h}$, while the shaded are corresponds to the true rectangle that generated the original labelling.

We compute a candidate $\hat{h}\colon \mathbb{R}^2 \to \{0, 1\}$ as the indicator function of the *smallest enclosing rectangle* of the point with label 1 (i.e., the blue points in Figure 6.1). It is clear that if we have *lots* of sampled points, then we should get a good approximation to the "true" rectangle that generated the data, while with few points this is not possible. How can we quantify this? Let $\epsilon > 0$ and $\delta \in (0, 1)$ be given, and let X be a random point with associated label $Y = f(X)$. Then $\mathbb{P}(f(X) = 1)$ is the probability measure of the true rectangle, while

$$R(\hat{h}) = \mathbb{P}(\hat{h}(X) \neq f(X))$$

is the risk of $\hat{h}$. We would like to find out the number of samples that would ensure

$$\mathbb{P}(R(\hat{h}) \leq \epsilon) \geq 1 - \delta.$$

First, note that since the rectangle defined by $\hat{h}$ is always contained in the true rectangle that we would like to discover, we can only get false negatives from $\hat{h}$ (that is, if

$\hat{h}(x) = 1$ then x is in the true rectangle, but there may be points x in the true rectangle for which $\hat{h}(x) = 0$). Hence,

$$\begin{aligned} R(\hat{h}) &= \mathbb{P}(\hat{h}(X) \neq f(X)) \\ &= \underbrace{\mathbb{P}(\hat{h}(X) = 1, f(X) = 0)}_{=0} + \mathbb{P}(\hat{h}(X) = 0, f(X) = 1) \\ &= \mathbb{P}(\hat{h}(X) = 0, f(X) = 1). \end{aligned}$$

If we denote by $\hat{B} = \{x \in \mathbb{R}^2 \colon \hat{h}(x) = 1\}$ the *computed* smallest enclosing rectangle, then the risk $R(\hat{h})$ can be described more geometrically as

$$R(\hat{h}) = \mathbb{P}(X \in B \backslash \hat{B}),$$

namely the probability of an input being in the true rectangle but not in the computed one.

Now let $\epsilon > 0$ and $\delta \in (0, 1)$ be given. If $\mathbb{P}(X \in B) \leq \epsilon$, then clearly also $R(\hat{h}) \leq \epsilon$. Assume therefore that $\mathbb{P}(X \in B) > \epsilon$. Denote by R_i, $i \in \{1, 2, 3, 4\}$, the smallest sub-rectangles or B with $\mathbb{P}(X \in R_i) \geq \epsilon/4$ that bound each of the four sides of B, respectively (see Figure 6.2). We could, for example, start with the whole rectangle and move one of its sides towards the opposite side for as long as the measure is not less than $\epsilon/4$.

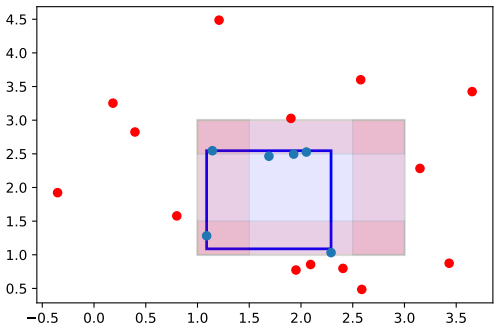

Figure 6.2: Four boundary regions with probability mass $\epsilon/4$ each.

Denote by R_i° the rectangles with their inward-facing sides removed. Then clearly the probability measure of the union of these sets is $\mathbb{P}(X \in \bigcup_i R_i^\circ) \leq \epsilon$, since the measure of each of the R_i° is at most $\epsilon/4$. If the computed rectangle $\hat{B}$ intersects all the R_i, then

$$\mathbb{P}(X \in B \backslash \hat{B}) = \mathbb{P}\left(X \in \bigcup_i R_i^\circ \backslash \hat{B}\right) \leq \epsilon.$$

We now need to show that the probability that $\hat{B}$ *does not* intersect all the rectangles is small:

$$\mathbb{P}(\exists i\colon \hat{B} \cap R_i = \emptyset) = \mathbb{P}\left(\bigcup_i \{\hat{B} \cap R_i = \emptyset\}\right) \leq \sum_{i=1}^{4} \mathbb{P}(\hat{B} \cap R_i = \emptyset),$$

where we used the union bound. The probability that $\hat{B}$ does not intersect one of the rectangles R_i, each of which has probability mass $\epsilon/4$, is equal to the probability that the n randomly sampled points that gave rise to $\hat{B}$ do not fall in R_i. For each of these points, the probability of *not* falling into R_i is at most $1 - \epsilon/4$, so the probability that none of the points falls into R_i is $(1 - \epsilon/4)^n$. Hence,

$$\mathbb{P}(\exists i\colon \hat{B} \cap R_i = \emptyset) \leq 4(1 - \epsilon/4)^n \leq 4\mathrm{e}^{-n\epsilon/4},$$

where we used the inequality $1 - x \leq \mathrm{e}^{-x}$. Setting the righ-hand side to δ, we conclude that if

$$n \geq \frac{4}{\epsilon} \log\left(\frac{4}{\delta}\right)$$

then $R(\hat{h}) \leq \epsilon$ holds with probability at least $1 - \delta$.

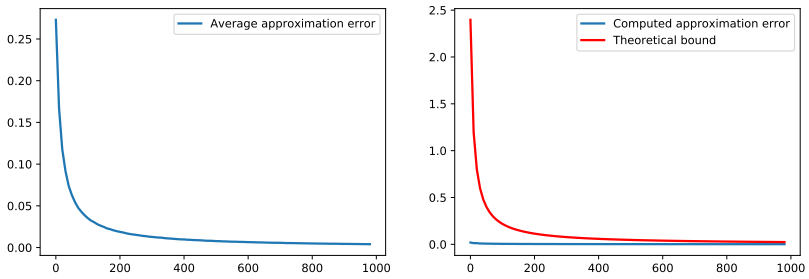

Figure 6.3: The left graphs shows the *average* risk as n increases. The right right graph shows the risk bound ϵ when given a confidence $1 - \delta = 0.99$ (blue curve), and the theoretical generalization bound as derived in the example.

Remark 6.3. Note that we did not make any assumptions on the probability distribution when deriving the bound on n in the rectangle-learning example. If the distribution is absolutely continuous with respect to the Lebesgue measure on $\mathbb{R}^2$, then we could have required the probability measures of the rectangles to be exactly $\epsilon/4$, but the way the proof is written it also applies to distributions that are not supported on all of $\mathbb{R}^2$, such as the uniform distribution on a compact subset of $\mathbb{R}^2$ that may or may not cover the area of B, or a discrete distribution supported on countably many points. The requirement $\mathbb{P}(X \in B) > \epsilon$ still ensures that enough probability mass is contained within the confines of B for the argument to work. We may, however, end up looking at degenerate cases where, for example, all the probability mass is on an

edge, one of the rectangles R_i is an edge or the whole of B, etc. Note that in such cases the intuitive view of the generalization risk as the "area" of the complement $B\backslash\hat{B}$ is no longer accurate! In practice we will only consider distributions that are natural to the situation under consideration.

Notes

The PAC framework is due to Warwick alumnus and Turing Award laureate Leslie Valiant [2]. Our presentation of the problem of learning rectangles is based on [1, Chapter 2].

[1] M. Mohri, A. Rostamizadeh, and A. Talwalkar. *Foundations of Machine Learning*. Adaptive Computation and Machine Learning series. MIT Press, 2012.

[2] Leslie G Valiant. A theory of the learnable. *Communications of the ACM*, 27(11):1134–1142, 1984.