Formulation of the learning problem, Part I

Maxim Raginsky

September 4, 2013

Now that we have seen an informal statement of the learning problem, as well as acquired some technical tools in the form of concentration inequalities, we can proceed to define the learning problem formally. Recall that the basic goal is to be able to predict some random variable Y of interest from a correlated random observation X, where the predictor is to be constructed on the basis of n i.i.d. training samples $(X_1, Y_1), \ldots, (X_n, Y_n)$ from the joint distribution of (X, Y). We will start by looking at an idealized scenario (often called the *realizable case* in the literature), in which Y is a *deterministic* function of X, and we happen to know the function class to which it belongs. This simple set-up will let us pose, in a clean form, the basic requirements a learning algorithm should satisfy. Once we are done with the realizable case, we can move on to the general setting, in which the relationship between X and Y is probabilistic and not known precisely. This is often referred to as the *model-free* or *agnostic* case.

This order of presentation is, essentially, historical. The first statement of the learning problem is hard to trace precisely, but the "modern" algorithmic formalization seems to originate with the 1984 work of Valiant [Val84] on learning Boolean formulae. Valiant has focused on *computationally efficient* learning algorithms. The agnostic (or model-free) formulation was first proposed and studied by Haussler [Hau92] in 1992.

In this lecture, I will be closely following the excellent exposition of Vidyasagar [Vid03, Ch. 3].

1 The realizable case

We start with an idealized scenario, now often referred to in the literature as the *realizable case*. The basic set-up is as follows. We have a set X (often called the *feature space* or *input space*) and a family $\mathscr{P}$ of probability distributions on X. We obtain an i.i.d. sample $X^n = (X_1, \ldots, X_n)$ drawn according to some $P \in \mathscr{P}$, which we do not know (although it may very well be the case that $\mathscr{P}$ is a singleton, $|\mathscr{P}| = 1$, in which case we, of course, *do* know P). We will look at two basic problems:

1. *Concept learning:* There is a class $\mathscr{C}$ of subsets of X, called the *concept class*, and an unknown *target concept* $C^* \in \mathscr{C}$ is picked by Nature. For each feature X_i in our sample X^n, we receive a binary *label* $Y_i = \mathbf{1}_{\{X_i \in C^*\}}$. The n feature-label pairs form the *training set*

$$(X_1, Y_1) = (X_1, \mathbf{1}_{\{X_1 \in C^*\}}), \ldots, (X_n, Y_n) = (X_n, \mathbf{1}_{\{X_n \in C^*\}}). \tag{1}$$

   The objective is to approximate the target concept C^* as accurately as possible.

2. *Function learning:* There is a class $\mathscr{F}$ of functions $f : \mathsf{X} \to [0, 1]$, and an unknown *target function* $f^* \in \mathscr{F}$ is picked by nature. For each input point X_i in the sample X^n, we receive a real-valued *output* $Y_i = f^*(X_i)$. The n input-output pairs

$$(X_1, Y_1) = (X_1, f^*(X_1)), \ldots, (X_n, f^*(X_n)). \tag{2}$$

The objective is to approximate the target function f^* as accurately as possible. (**Note:** the requirement that f map X into $[0,1]$ is imposed primarily for technical convenience; using appropriate moment and/or tail behavior assumptions on P, it is possible to remove this requirement, but the resulting proofs will be somewhat laborious.)

We will now consider these two problems separately.

1.1 Concept learning

As we already stated, the goal of concept learning is to approximate the target concept C^* as accurately as possible on the basis of the training data (1). This is done by means of a *learning algorithm.* An algorithm of this sort should be capable of producing an approximation to C^* given the training set of the form (1) of any size n. More precisely:

Definition 1. *A* concept learning problem *is specified by a triple* $(\mathsf{X}, \mathscr{P}, \mathscr{C})$, *where* X *is the feature space,* $\mathscr{P}$ *is a family of probability distributions on* X, *and* $\mathscr{C}$ *is a concept class. A* learning algorithm *for* $(\mathsf{X}, \mathscr{P}, \mathscr{C})$ *is a sequence* $\mathscr{A} = \{A_n\}_{n=1}^{\infty}$ *of mappings*

$$A_n : (\mathsf{X} \times \{0,1\})^n \to \mathscr{C}.$$

If $\mathscr{P}$ consists of only one distribution P, then the mappings A_n may depend on P; otherwise, they may only depend on $\mathscr{P}$ as a whole. The idea behind the above definition is that for each training set size n we have a definite procedure for forming an approximation to the unknown target concept C^* on the basis of the training set of that size.

For brevity, let us denote by Z_i the ith training pair $(X_i, Y_i) = (X_i, \mathbf{1}_{\{X_i \in C^*\}})$, and let us denote by Z the set $\mathsf{X} \times \{0,1\}$. Given a training set $Z^n = (Z_1, \dots, Z_n) \in \mathsf{Z}^n$ and a learning algorithm $\mathscr{A}$, the approximation to C^* is

$$\widehat{C}_n = A_n(Z^n) = A_n(Z_1, \dots, Z_n) = A_n\left((X_1, \mathbf{1}_{\{X_1 \in C^*\}}), \dots, (X_n, \mathbf{1}_{\{X_n \in C^*\}})\right).$$

Note that $\widehat{C}_n$ is an element of the concept class $\mathscr{C}$ (by definition), and that it is a random variable since it depends on the random sample Z^n. It is often referred to as a *hypothesis* output by the learning algorithm $\mathscr{A}$.

How shall we measure the goodness of this approximation $\widehat{C}_n$? A natural thing to do is the following. Suppose now we draw a fresh feature X from the same distribution $P \in \mathscr{P}$ as the one that has generated the training feature set X^n and venture a *hypothesis* that X belongs to the target concept C^* if $X \in \widehat{C}_n$, i.e., if $\mathbf{1}_{\{X \in \widehat{C}_n\}} = 1$. When would we make a mistake, i.e., *misclassify* X? There are two mutually exclusive cases:

1. X is in C^*, but not in $\widehat{C}_n$, i.e., $X \in C^* \cap \widehat{C}_n^c$, where $\widehat{C}_n^c = \mathsf{X} \backslash \widehat{C}_n$ is the complement of $\widehat{C}_n$ in X.
2. X is not in C^*, but it is in $\widehat{C}_n$, i.e., $X \in (C^*)^c \cap \widehat{C}_n$.

Thus, we will misclassify X precisely when it happens to lie in the *symmetric difference*

$$C^* \triangle \widehat{C}_n \triangleq (C^* \cap \widehat{C}_n^c) \cup ((C^*)^c \cap \widehat{C}_n).$$

This will happen with probability $P(C^* \triangle \widehat{C}_n)$ — note, by the way, that this is a random number since $\widehat{C}_n$ depends on the training data Z^n. At any rate, we take the P-probability of the symmetric difference $C^* \triangle \widehat{C}_n$ as our measure of performance of $\mathscr{A}$. In order to streamline the notation, let us define the *risk* (or *loss*) of any $C \in \mathscr{C}$ w.r.t. C^* and P as

$$L_P(C, C^*) \triangleq P(C \triangle C^*) = P(X \in C \triangle C^*).$$

Exercise 1. *Prove that*

$$L_P(C,C^*) = \int_{\mathsf{X}} \left|\mathbf{1}_{\{x\in C\}} - \mathbf{1}_{\{x\in C^*\}}\right|^2 P(\mathrm{d}x).$$

In other words, $L_P(C,C^*)$ is the squared $L^2(P)$ norm of the difference of the indicator functions $I_C(\cdot) = \mathbf{1}_{\{\cdot\in C\}}$ and $I_{C^*}(\cdot) = \mathbf{1}_{\{\cdot\in C^*\}}$, $L_P(C,C^*) = \|I_C - I_{C^*}\|^2_{L^2(P)}$.

Roughly speaking, we will say that $\mathscr{A}$ is a good algorithm if

$$L_P(\widehat{C}_n, C^*) \to 0 \qquad \text{as } n \to \infty \tag{3}$$

for any $P \in \mathscr{P}$ and any $C^* \in \mathscr{C}$. Since $\widehat{C}_n$ is a random element of $\mathscr{C}$, the convergence in (3) can only be in some probabilistic sense. In order to make things precise, for any $C \in \mathscr{C}$ let us denote by P_C the joint distribution of a pair $Z = (X,Y)$, where $X \sim P$ and $Y = \mathbf{1}_{\{X\in C\}}$. Then we define the following two quantities:

$$r_{\mathscr{A}}(n,\varepsilon,P) \triangleq \sup_{C\in\mathscr{C}} P^n_C\left(Z^n \in \mathsf{Z}^n : L_P(A_n(Z^n),C) \geq \varepsilon\right)$$
$$\bar{r}_{\mathscr{A}}(n,\varepsilon,\mathscr{P}) \triangleq \sup_{P\in\mathscr{P}} r_{\mathscr{A}}(n,\varepsilon,P)$$

where P^n_C denotes the n-fold product of P. For a fixed P (which amounts to assuming that the features X^n were drawn i.i.d. from P), $r_{\mathscr{A}}(n,\varepsilon,P)$ quantifies the worst-case "size" of the set of "bad" samples, where we say that a sample X^n is bad if it causes the learning algorithm $\mathscr{A}$ to output a hypothesis $\widehat{C}_n = A_n(Z^n)$ whose risk is larger than ε. The worst case is over the entire concept class $\mathscr{C}$, since we do not know the target concept C^*. The quantity $\bar{r}_{\mathscr{A}}(n,\varepsilon,\mathscr{P})$ accounts for the fact that we do not know which $P \in \mathscr{P}$ has generated the training feature points.

With all these things defined, we can now state the following:

Definition 2. *A learning algorithm $\mathscr{A} = \{A_n\}$ is* probably approximately correct *(or* PAC*)* to accuracy ε *if*

$$\lim_{n\to\infty} \bar{r}_{\mathscr{A}}(n,\varepsilon,\mathscr{P}) = 0. \tag{4}$$

We say that $\mathscr{A}$ is PAC if it is PAC to accurary ε for every $\varepsilon > 0$. The concept class $\mathscr{C}$ is called PAC learnable to accuracy ε w.r.t. $\mathscr{P}$ *if there exists an algorithm that is PAC to accuracy ε. Finally, we say that $\mathscr{C}$ is* PAC learnable *if there exists an algorithm that is PAC.*

The term "probably approximately correct," which seems to have first been introduced by Angluin [Ang88], is motivated by the following observations. First, the hypothesis $\widehat{C}_n$ output by $\mathscr{A}$ for some n is only an *approximation* to the target concept C^*. Thus, $L_P(\widehat{C}_n, C^*)$ will be, in general, nonzero. But if it is small, then we are justified in claiming that $\widehat{C}_n$ is *approximately correct*. Secondly, we may always encounter a bad sample, so $L_P(\widehat{C}_n, C^*)$ can be made small only *with high probability*. Thus, informally speaking, a PAC algorithm is one that "works reasonably well most of the time."

An equivalent way of phrasing the statement that a learning algorithm is PAC is as follows: For any $\varepsilon > 0$ and $\delta > 0$, there exists some $n(\varepsilon,\delta) \in \mathbb{N}$, such that

$$P^n_C\left(Z^n \in \mathsf{Z}^n : L_P(A_n(Z^n),C) \geq \varepsilon\right) \leq \delta, \qquad \forall n \geq n(\varepsilon,\delta), \forall C \in \mathscr{C}, \forall P \in \mathscr{P}. \tag{5}$$

In this context, ε is called the *accuracy parameter*, while δ is called the *confidence parameter*. The meaning of this alternative characterization is as follows. If the sample size n is at least $n(\varepsilon,\delta)$, then we can state with confidence at least $1-\delta$ that the hypothesis $\widehat{C}_n$ will correctly classify a fresh random point $X \in \mathsf{X}$ with probability at least $1-\varepsilon$.

The two problems of interest to us are:

1. Determine conditions under which a given concept class $\mathscr{C}$ is PAC learnable.
2. Obtain upper and lower bounds on $n(\varepsilon,\delta)$ as a function of ε,δ. The following terminology is often used: the smallest number $n(\varepsilon,\delta)$ such that (5) holds is called the *sample complexity*.

1.2 Function learning

The goal of function learning is to construct an accurate approximation to an unknown target function $f^* \in \mathscr{F}$ on the basis of training data of the form (2). Analogously to the concept learning scenario, we have:

Definition 3. *A* function learning problem *is specified by a triple* $(\mathsf{X},\mathscr{P},\mathscr{F})$*, where* X *is the input space,* $\mathscr{P}$ *is a family of probability distributions on* X*, and* $\mathscr{F}$ *is a class of functions* $f : \mathsf{X} \to [0,1]$*. A* learning algorithm *for* $(\mathsf{X},\mathscr{P},\mathscr{F})$ *is a sequence* $\mathscr{A} = \{A_n\}_{n=1}^{\infty}$ *of mappings*

$$A_n : (\mathsf{X} \times [0,1])^n \to \mathscr{F}.$$

As before, let us denote by Z_i the input-output pair $(X_i, Y_i) = (X_i, f^*(X_i))$ and by Z the product set $\mathsf{X} \times [0,1]$. Given a training set $Z^n = (Z_1,\dots,Z_n) \in \mathsf{Z}^n$ and a learning algorithm $\mathscr{A}$, the approximation to f^* is

$$\widehat{f}_n = A_n(Z^n) = A_n\big((X_1, f^*(X_1)),\dots,(X_n, f^*(X_n))\big).$$

As in the concept learning setting, $\widehat{f}_n$ is a *random element* of the function class $\mathscr{F}$.

In order to measure the performance of $\mathscr{A}$, we again imagine drawing a fresh input point $X \in \mathsf{X}$ from the same distribution $P \in \mathscr{P}$ that has generated the training inputs X^n. A natural error metric is the squared loss $|\widehat{f}_n(X) - f^*(X)|^2$. As before, we can define the *risk* (or *loss*) of any $f \in \mathscr{F}$ w.r.t. f^* and P as

$$L_P(f, f^*) \triangleq \mathbb{E}_P|f(X) - f^*(X)|^2 = \|f - f^*\|^2_{L^2(P)} = \int_{\mathsf{X}} |f(x) - f^*(x)|^2 P(\mathrm{d}x). \tag{6}$$

Thus, the quantity of interest is the risk of $\widehat{f}_n$:

$$L_P(\widehat{f}_n, f^*) = \int_{\mathsf{X}} |\widehat{f}_n(x) - f^*(x)|^2 P(\mathrm{d}x).$$

Keep in mind that $L_P(\widehat{f}_n, f^*)$ is a random variable, as it depends on $\widehat{f}_n$, which in turn depends on the random sample $Z^n \in \mathsf{Z}^n$.

Remark 1. The concept learning problem is, in fact, a special case of the function learning problem. Indeed, fix a concept class $\mathscr{C}$ and consider the function class $\mathscr{F}$ consisting of the indicator functions of the sets in $\mathscr{C}$:

$$\mathscr{F} = \{I_C : C \in \mathscr{C}\}.$$

Then for any $f = I_C$ and $f^* = I_{C^*}$ we will have

$$L_P(f, f^*) = \|I_C - I_{C^*}\|^2_{L^2(P)} = P(C \triangle C^*),$$

which is the error metric we have defined for concept learning.

If for each $f \in \mathscr{F}$ we denote by P_f the joint distribution of a pair $Z = (X, Y)$, where $X \sim P$ and $Y = f(X)$, then for a given learning problem $(\mathsf{X}, \mathscr{P}, \mathscr{F})$ and a given algorithm $\mathscr{A}$ we can define

$$\begin{aligned} r_{\mathscr{A}}(n, \varepsilon, P) &\triangleq \sup_{f \in \mathscr{F}} P_f^n\left(Z^n \in \mathsf{Z}^n : L_P(A_n(Z^n), f) \geq \varepsilon\right) \\ \bar{r}_{\mathscr{A}}(n, \varepsilon, \mathscr{P}) &\triangleq \sup_{P \in \mathscr{P}} r_{\mathscr{A}}(n, \varepsilon, P) \end{aligned}$$

for every $n \in \mathbb{N}$ and $\varepsilon > 0$.The meaning of these quantities is exactly parallel to the corresponding quantities in concept learning, and leads to the following definition:

Definition 4. *A learning algorithm* $\mathscr{A} = \{A_n\}$ *is* PAC to accuracy ε *if*

$$\lim_{n \to \infty} \bar{r}_{\mathscr{A}}(n, \varepsilon, \mathscr{P}) = 0,$$

and PAC if it is PAC to accuracy ε *for all* $\varepsilon > 0$. *A function class* $\mathscr{F} = \{f : \mathsf{X} \to [0,1]\}$ *is* PAC-learnable (to accuracy ε) *w.r.t.* $\mathscr{P}$ *if there exists an algorithm* $\mathscr{A}$ *that is PAC for* $(\mathsf{X}, \mathscr{P}, \mathscr{F})$ *(to accuracy* ε*).*

An equivalent way of stating that $\mathscr{A}$ is PAC is that, for any $\varepsilon, \delta > 0$ there exists some $n(\varepsilon, \delta) \in \mathbb{N}$ such that

$$P^n\left(Z^n \in \mathsf{Z}^n : L_P(A_n(Z^n), f) \geq \varepsilon\right) \leq \delta, \qquad \forall n \geq n(\varepsilon, \delta), \forall f \in \mathscr{F}, \forall P \in \mathscr{P}.$$

The smallest $n(\varepsilon, \delta) \in \mathbb{N}$ for which the above inequality holds is termed the *sample complexity*.

2 Example: learning axis-parallel rectangles

To make these ideas concrete, let us consider an example of a PAC-learnable concept class. Given what we know at this point, the only way to show that a given concept class is PAC-learnable is to exhibit an algorithm which is PAC. Later on, we will develop generic tools that will allow us to determine PAC-learnability without having to construct an algorithm for each separate case.

We take $\mathsf{X} = [0,1]^2$, the unit square in the plane, let $\mathscr{P}$ be the class of all probability distributions on X (w.r.t. the usual Borel σ-algebra, but let's not worry about that), and let $\mathscr{C}$ be the collection of all *axis-parallel rectangles*: that is, a set C is in $\mathscr{C}$ if and only if it is of the form

$$\begin{aligned} C &= [a_1, b_1] \times [a_2, b_2] \\ &= \left\{(x_1, x_2) \in [0,1]^2 : a_1 \leq x_1 \leq b_1,\, a_2 \leq x_2 \leq b_2\right\} \end{aligned}$$

for some $0 \leq a_1 \leq b_1 \leq 1$ and $0 \leq a_2 \leq b_2 \leq 1$ (see Figure 1).

We now describe our learning algorithm. Given a training set $Z^n = \left(Z_1, \ldots, Z_n\right) = \left((X_1, Y_1), \ldots, (X_n, Y_n)\right)$, we say that the ith training pair $Z_i = (X_i, \mathbf{1}_{\{X_i \in C^*\}})$ is a *positive example* if $Y_i = 1$ (i.e., if X_i belongs to the target concept C^*), and is a *negative example* otherwise. Our algorithm $\mathscr{A} = \{A_n\}_{n=1}^{\infty}$ is the following intuitive rule: for each n, we take

$$\widehat{C}_n = A_n(Z^n) = \text{smallest rectangle } C \in \mathscr{C} \text{ that contains all positive examples in } Z^n. \tag{7}$$

Figure 1 shows a particular instance of this algorithm. We will now prove the following result, originally due to Blumer et al. [BEHW89]:

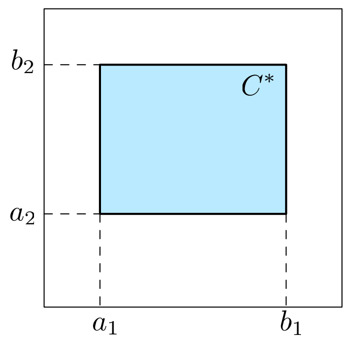

Figure 1: An axis-parallel rectangle.

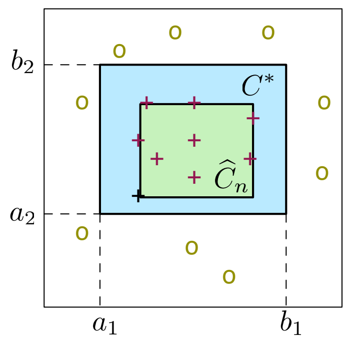

Figure 2: The hypothesis $\widehat{C}_n$ produced by algorithm (7). Positive (resp., negative) examples are shown as crosses (resp., zeros).

Theorem 1. *The algorithm $\mathscr{A}$ defined in (7), i.e., the one that returns the smallest axis-parallel rectangle that encloses all positive examples in Z^n, satisfies*

$$\bar{r}_{\mathscr{A}}(n, \varepsilon, \mathscr{P}) \le 4(1 - \varepsilon/4)^n. \tag{8}$$

Therefore, this algorithm is PAC, and the class $\mathscr{C}$ is PAC-learnable.

Proof. Since no positive example can lie outside C^*, the hypothesis $\widehat{C}_n$ produced by the algorithm (7) must lie inside C^*: $\widehat{C}_n \subseteq C^*$. Therefore,

$$\widehat{C}_n \triangle C^* = C^* \cap (\widehat{C}_n)^c \equiv C^* \setminus \widehat{C}_n. \tag{9}$$

If $P(C^*) < \varepsilon$, then from (9) it follows that $L_P(\widehat{C}_n, C^*) = P\big(C^* \setminus \widehat{C}_n\big) \le P(C^*) < \varepsilon$. Thus, we will assume that $P(C^*) \ge \varepsilon$. Suppose that $C^* = [a_1, b_1] \times [a_2, b_2]$ and $\widehat{C}_n = [\widehat{a}_1, \widehat{b}_1] \times [\widehat{a}_2, \widehat{b}_2]$, and consider the following four rectangles:

$$V_1 = [a_1, \widehat{a}_1] \times [a_2, b_2], \quad V_2 = [\widehat{b}_1, b_1] \times [a_2, b_2], \quad H_1 = [a_1, b_1] \times [a_2, \widehat{a}_2], \quad H_2 = [a_1, b_1] \times [\widehat{b}_2, b_2]$$

(see Figure 3). From (9), we see that the symmetric difference $\widehat{C}_n \triangle C^*$ is exactly equal to the union of

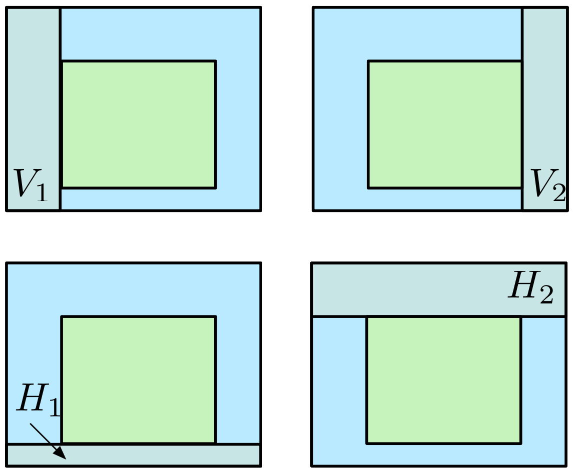

Figure 3: The four rectangles V_1, V_2, H_1, H_2 used in the proof of Theorem 1. The target concept C^* is in blue, the hypothesis $\widehat{C}_n$ returned by our algorithm is green.

these four rectangles, which we will denote by E:

$$\widehat{C}_n \triangle C^* = E \triangleq V_1 \cup V_2 \cup H_1 \cup H_2.$$

(Note, by the way, that V_1, V_2, H_1, H_2, E are all *random* rectangles, since they depend on the hypothesis $\widehat{C}_n$ and thus on the random training set Z^n.) Our goal is show that $P(E) \le \varepsilon$ with high probability.

We claim that, with probability at least $1 - 4(1-\varepsilon/4)^n$, each of the rectangles V_1, V_2, H_1, H_2 has P-probability of no more than $\varepsilon/4$. If that is the case, then $P(E) = P(V_1 \cup V_2 \cup H_1 \cup H_2) \le \varepsilon$ with probability at least $1 - 4(1-\varepsilon/4)^n$. From this, we conclude that, for any C^*,

$$P^n_{C^*}\left(Z^n \in \mathsf{Z}^n : L_P(A_n(Z^n), C^*) \ge \varepsilon\right) \le P^n_{C^*}\Big(Z^n \in \mathsf{Z}^n : P(E) \ge \varepsilon\Big) \le 4(1-\varepsilon/4)^n. \tag{10}$$

Since this bound holds for all C^* and for all $P \in \mathscr{P}$, we get (8).

It now remains to prove the claim. To that end, let A_1 be the smallest rectangle of the form $[a_1, a] \times [a_2, b_2]$, such that $P(A_1) \ge \varepsilon/4$, and choose A_2, B_1, B_2 in an analogous fashion. Consider the event that $P(V_1) \ge \varepsilon/4$. Recall our definition of A_1 as the *smallest* rectangle of the form $[a_1, a] \times [a_2, b_2]$ whose P-probability is at least $\varepsilon/4$. Then it must be the case that $A_1 \subseteq V_1$. Now, if $A_1 \subseteq V_1$, then there are no training examples inside A_1: there are no positive examples in A_1, since otherwise they would have been included into $\widehat{C}_n$, and there are no negative examples, since there can't be any in C^*. The probability of the event that there are no training examples in A_1 can be computed

$$\begin{aligned} \mathbb{P}\left(\bigcap_{i=1}^n \{X_i \not\in A_1\}\right) &= \prod_{i=1}^n \mathbb{P}(X_i \not\in A_1) && (11)\\ &= [\mathbb{P}(X \not\in A_1)]^n && (12)\\ &\le (1-\varepsilon/4)^n, && (13) \end{aligned}$$

where (11) is by independence of the X_i's, (12) follows from the fact that the X_i's are identically distributed, and (13) follows from the fact that $P(A_1) \geq \varepsilon/4$ by construction. Therefore,

$$\mathbb{P}\Big[P(V_1) \geq \varepsilon/4\Big] \leq \mathbb{P}\Big[X_i \not\in A_1, \forall i\Big] \leq (1-\varepsilon/4)^n.$$

Similar reasoning applies to A_2, B_1, B_2. Thus, with probability at least $1-4(1-\varepsilon/4)^n$,

$$P(V_1) \leq \varepsilon/4, \quad P(V_2) \leq \varepsilon/4, \quad P(H_1) \leq \varepsilon/4, \quad P(H_2) \leq \varepsilon/4,$$

and the claim is proved. □

Corollary 1. *The sample complexity of learning axis-parallel rectangles satisfies*

$$n(\varepsilon,\delta) \geq \frac{4\log(4/\delta)}{\varepsilon}. \tag{14}$$

Proof. From (8), $\bar{r}_{\mathscr{A}}(n,\varepsilon,\mathscr{P}) \leq \delta$ for all n such that $4(1-\varepsilon/4)^n \leq \delta$ or, equivalently, for all n, such that

$$n\log(1-\varepsilon/4) \leq \log(\delta/4). \tag{15}$$

Using the inequality $\log x \leq x-1$, we see that if n satisfies (14), then it will satisfy (15). (Of course, (15) is tighter, but all we wanted to see that the number of training examples sufficient to learn an unknown concept $C^* \in \mathscr{C}$ with accuracy ε and confidence $1-\delta$ is polynomial in $1/\varepsilon$ and polylogarithmic in $1/\delta$.) □

References

[Ang88] D. Angluin. Queries and concept learning. *Machine Learning*, 2:319–342, 1988.

[BEHW89] A. Blumer, A. Ehrenfeucht, D. Haussler, and M. K. Warmuth. Learnability and the Vapnik–Chervonenkis dimension. *Journal of the ACM*, 36(4):929–965, 1989.

[Hau92] D. Haussler. Decision theoretic generalizations of the PAC model for neural net and other learning applications. *Information and Computation*, 95:129–161, 1992.

[Val84] L. G. Valiant. A theory of the learnable. *Communications of the ACM*, 27(11):1134–1142, 1984.

[Vid03] M. Vidyasagar. *Learning and Generalization*. Springer, 2 edition, 2003.