Efficient Distribution-free Learning of Probabilistic Concepts

Michael J. Kearns Robert E. Schapire

AT&T Bell Laboratories
600 Mountain Avenue
P.O. Box 636
Murray Hill, New Jersey 07974-0636

August 25, 1993

Abstract

In this paper we investigate a new formal model of machine learning in which the concept (boolean function) to be learned may exhibit uncertain or probabilistic behavior—thus, the same input may sometimes be classified as a positive example and sometimes as a negative example. Such *probabilistic concepts* (or *p-concepts*) may arise in situations such as weather prediction, where the measured variables and their accuracy are insufficient to determine the outcome with certainty. We adopt from the Valiant model of learning [27] the demands that learning algorithms be efficient and general in the sense that they perform well for a wide class of p-concepts and for any distribution over the domain. In addition to giving many efficient algorithms for learning natural classes of p-concepts, we study and develop in detail an underlying theory of learning p-concepts.

1 Introduction

Consider the following scenarios:

A meteorologist is attempting to predict tomorrow's weather as accurately as possible. He measures a small number of presumably relevant parameters, such as the current temperature, barometric pressure, and wind speed and direction. He then makes a forecast of the form "chances for rain tomorrow are 70%." The next day it either rains or it does not rain.

A statistician wishes to compile an approximate rule for predicting when students will be admitted to a particular college. There are some students whose records are so strong they will be accepted regardless of which admissions officer reviews their files; similarly, there are others who are categorically rejected. For many students, however, their admission may be highly dependent on the particular admissions officer that evaluates their applications; thus the best model for the chances of these borderline students involves a probability of acceptance. However, every student is either accepted or rejected.

This research was conducted while the authors were at the Massachusetts Institute of Technology Laboratory for Computer Science.

Authors' network addresses: {mkearns, schapire}@research.att.com.

A physicist is attempting to determine the orientation of spin for particles in a certain magnetic field. Presumably, the orientation of spin is at least partially determined by a genuinely random process of Nature. The spin of any particle is always oriented either up or down.

We wish to produce a good model for the recognition of common objects such as chairs. For most objects in the world, there is nearly universal agreement as to whether that object is a chair or a non-chair. There do exist, however, a few objects that may provoke disagreement, such as stools and benches. This is due to the fact that the concept of "chair" is not absolute, and semantic boundaries of this concept may be exposed by both naturally occurring and artificially constructed objects. Most young children, however, are not explicitly told about such definitional shortcomings; they are simply told whether or not something is a chair.

There are some obvious common themes in each of the above situations. First, in each there is *uncertain* or probabilistic behavior. This uncertainty may arise for rather different reasons. For example, in the case of the meteorologist, it could be that while the weather is in principle a deterministic process, the parameters measured by the meteorologist and the limited accuracy of these measurements are insufficient to determine this process. In the case of the physicist, the electron spin is believed to be governed to some degree by an ideal random process. In the case of the statistician, the uncertainty arises from the diversity of human behavior, and in the case of chair recognition, a probability may model the difficulties of providing a perfectly precise definition to an inherently uncertain or "fuzzy" concept.

A second theme that is common to each of these settings is the fact that even though the best model may be a conditional probability $c(x)$ that the event (rain, acceptance to the college, etc.) occurs given x (where x represents the measured weather variables or a student's application), the observer only witnesses whether or not the event occurs. Thus, examples are of the form $(x, 0)$ or $(x, 1)$ — *not* $(x, c(x))$ — and the $\{0, 1\}$ label provided with x is distributed according to the conditional probability $c(x)$. Furthermore, we should not expect to be able to compute even an *estimate* of $c(x)$ from the given $\{0, 1\}$-labeled examples, since in general we are unlikely to ever see the same x twice (each day's weather is at least slightly different, as is each student's application).

Finally, although there is uncertainty in each of these settings, there is also some *structure* to this uncertainty. For instance, days with nearly identical atmospheric conditions and students with very similar high school records can be expected to have nearly equal probabilities of rain and acceptance to the college, respectively. We also expect some inputs to be assigned conditional probabilities that are very near 0 or 1; for example, days on which the sky is cloudless, or students with straight A's. This structured behavior strongly distinguishes these learning scenarios from a "noisy" setting, such as the one considered by Angluin and Laird [3], Kearns and Li [16], and Sloan [26]. In a model of learning with noise, the noise is typically "white" (that is, all inputs have either an equal probability of corruption or a probability determined by an adversary), and the noise is regarded as something an algorithm wishes to "filter out" in an attempt to uncover some underlying *deterministic* concept. In the examples given above, the probabilistic behavior is both *structured* (possibly in a manner that can be exploited by a learning algorithm) and *inherently part* *of the underlying phenomenon*. Thus, whenever possible we do not wish to filter this probabilistic behavior out of the hypothesis, but rather to *model* it.

In this paper we wish to study a model of learning in such uncertain environments. We formalize these settings by introducing the notion of a *probabilistic concept* (or *p-concept*). A p-concept c over a domain set X is simply a mapping $c : X \to [0, 1]$. For each $x \in X$, we interpret $c(x)$ as the probability that x is a positive example of the p-concept c. Following the discussion above, a learning algorithm in this framework is attempting to infer something about the underlying target p-concept c solely on the basis of labeled examples (x, b), where $b \in \{0, 1\}$ is a bit generated randomly according to the conditional probability $c(x)$, i.e., $b = 1$ with probability $c(x)$.

The value $c(x)$ may be viewed as a measure of the degree to which x exemplifies some concept c. In this sense, p-concepts are quite similar to the related notion of a *fuzzy set*, a kind of "set" whose boundaries are fuzzy or unclear, and whose formal definition is nearly identical to that of a p-concept. An axiomatic theory of fuzzy sets was introduced by Zadeh [32], and they have since received much treatment by researchers in the field of pattern recognition. (See Kandel's book [14] for a good introduction.)

We distinguish two possible goals for a learning algorithm in the p-concept model. The first and easier goal is that of *label prediction*: the algorithm wishes to output a hypothesis that maximizes the probability of correctly predicting the $\{0, 1\}$ label generated by c on an input x. We call this kind of learning *decision-rule learning*, since we are not primarily concerned with actually modeling the underlying uncertainty but instead wish to accurately predict the observable $\{0, 1\}$ outcome of this uncertainty. We will see that the more difficult and more interesting goal is that of finding a good *model of probability*. Here the algorithm wishes to output a hypothesis p-concept $h : X \to [0, 1]$ that is a good real-valued approximation to the target c; thus, we want $|c(x) - h(x)|$ to be small for most inputs x. Following the motivation given above, we are mainly concerned with this latter notion of learning.

As mentioned above, the quantity $c(x)$ is simply the conditional probability of the label 1 being assigned to a given instance x. Equivalently, in this setting, $c(x)$ is the conditional expectation of the label b assigned to the given instance x. Thus, the problem of learning a model of probability can be described in statistical terms as that of modeling the conditional distribution of some random variable b (the label) given the value of some other random variable x (the instance). This problem, commonly known to statisticians as *regression*, has received much attention in the statistics literature. (See, for instance, Dobson's book [8].) The problem of learning a model of probability is also equivalent (with slight restrictions) to that of learning a *stochastic rule* as defined in the parallel work of Yamanishi [31].

As noted above, we will typically assume that the probabilistic behavior exhibited by the target p-concept is, to some degree, structured. To model this structure, we study the learnability of classes of p-concepts that obey natural mathematical properties intended to model some realistic environments. As a simple example, in constructing a p-concept model of the subjective notion of "tall," it is reasonable to assume that $x \geq y$ implies $c(x) \geq c(y)$ (where x represents height) — the taller a person actually is, the higher the percentage of people who will agree he is tall (or the greater the "degree of tallness" we wish to assign). This motivates us to consider learning the class $\mathcal{C}$ of all non-decreasing p-concepts over the positive real line. In general, we wish to study the learnability of p-concept classes that are restricted in such a way as to plausibly capture some realistic situation, but are not so restricted as to make the learning problem trivial or uninteresting.

We adopt from the Valiant model for learning deterministic concepts [27] the emphasis on learning algorithms that are both efficient (in the sense of polynomial time) and general (in the sense of working for the largest possible p-concept classes and against any probability distribution over the domain). After formalizing the learning model and the two possible goals for a learning algorithm (decision-rule learning and model-of-probability learning), we embark on a systematic study of techniques for designing efficient algorithms for learning p-concepts and the underlying theory of the p-concept model.

We begin by giving examples of efficient algorithms producing a good model of probability that employ what we call the *direct approach*; the analyses of these algorithms give first-principles arguments that the output hypothesis is good. These include algorithms for arbitrary non-decreasing functions motivated above, a probabilistic analog of Rivest's decision lists [24], and a class of "hidden-variable" p-concepts, motivated by settings such as weather prediction where the apparently probabilistic behavior may in part be due to the fact that some relevant quantities remain undiscovered.

We then consider the problem of *hypothesis testing* in the p-concept model. Working in the same framework as Haussler [11, **?**], we define a *loss function* that assigns a measure of goodness to any hypothesis p-concept on a $\{0,1\}$-labeled sample. After observing that the *quadratic loss* measure has some well-studied mathematical properties that make it a convenient and appropriate choice for our setting, we next give an example of an efficient algorithm for finding a model of probability that first does some direct computation to narrow the search and then uses quadratic loss to choose the best hypothesis among a small remaining pool. This algorithm learns a class of p-concepts in which only a small number of variables are relevant, but the dependence on these variables may be arbitrary.

Next we consider the related but more difficult issue of *uniform convergence* of a p-concept class. More precisely, how many $\{0,1\}$-labeled examples must be taken before we have high confidence that every p-concept in the class has an empirical quadratic loss that accurately reflects its true performance as a model of probability? In a more general formulation, this question has received extensive consideration in the statistical pattern recognition literature, and its importance to learning has been demonstrated by many recent papers. We show that the sufficient sample size for uniform convergence is bounded above by the *pseudo dimension* of the p-concept class, a combinatorial measure discussed by Pollard [23], Haussler [**?**] and other authors.

We then give efficient algorithms that apply the uniform convergence method (that is, take a large enough sample as dictated by the quadratic loss dimension, and find the hypothesis minimizing the empirical loss over the sample) in order to find a good model of probability. In particular, we prove the effectiveness of an algorithm for learning p-concepts represented by linear combinations of d given basis functions. We then show that the quadratic loss dimension, when finite, is also a lower bound on the required sample size for learning any p-concept class with a model of probability; thus the quadratic loss dimension, when finite, characterizes the sample complexity of p-concept learning with a model of probability in the same way that the Vapnik-Chervonenkis (VC) dimension characterizes sample complexity in Valiant's model. (See Blumer et al.'s paper [6] for a full discussion of the VC-dimension.) However, we show that p-concept classes of *infinite* quadratic loss dimension may sometimes be learned efficiently, in contrast to classes of infinite VC-dimension in the Valiant model, which are not learnable in *any* amount of time. (Technically, this is not always true if "dynamic" sampling is allowed; see Linial, Mansour and Rivest's paper [19] for further details.)

We conclude with an investigation of Occam's Razor in the p-concept model. In the Valiant model, Blumer et al. [5] show that it suffices for learning to find a consistent hypothesis that is slightly shorter than the sample data. We look for analogies in our setting: namely, when does "data compression" imply a good model of probability? We formalize this question, and argue briefly that several of our algorithms can be interpreted as implementing a form of data compression.

The primary contribution of this research is that of providing initial positive results for efficient learnability in a natural and important extension to Valiant's model. This may be significant because the Valiant model has been criticized for its strong hardness results and drought of powerful positive results, as well as for the unrealistic deterministic and noise-free view it takes of the concepts to be learned.

At first, it may seem paradoxical that we are able to simultaneously generalize the model and obtain several positive results; perhaps this can be intuitively explained by the fact that since we generalize the form of the representations being learned, there are more ways that concepts capturing some natural and realistic setting may be simply expressed. In contrast, since the Valiant model tends to emphasize concept classes based on standard circuit complexity, one is quickly led to study very powerful and apparently difficult classes such as disjunctive normal form Boolean expressions.

Another contribution of this research is in demonstrating the feasibility and practicality of the approach suggested by Haussler [11]. His work addressed the issue of sample complexity upper bounds in great generality, even encompassing the case where the input-output relation to be learned has no prescribed functional form. This generality prevents Haussler from obtaining either good sample size lower bounds or efficient learning algorithms; indeed, he cites both of these as important areas for further research. Our results may be regarded as a first demonstration of applying some of Haussler's general principles to a specific and realistic model in which computation time is of foremost significance.

2 The learning model

Let X be a set called the *domain* (or *instance space*). A *probabilistic concept* (or *p-concept*) is a real-valued function $c : X \to [0, 1]$. When learning the p-concept c, the value $c(x)$ is interpreted as the *probability* that x exemplifies the concept being learned (i.e., the probability that x is a positive example). A *p-concept class* $\mathcal{C}$ is a family of p-concepts. On any execution, a learning algorithm for $\mathcal{C}$ is attempting to learn a distinguished *target* p-concept $c \in \mathcal{C}$ with respect to a fixed but unknown and arbitrary *target distribution* D over X. We think of D as modeling the natural distribution of objects in the domain, and c represents the probabilistic concept to be learned in this domain. (More formally, D is a probability measure on a σ-algebra of measurable subsets of X. We assume implicitly that all of the p-concepts considered are measurable functions with respect to this σ-algrebra on X, and the Borel σ-algebra on $[0,1]$.)

The learning algorithm is given access to an oracle *EX* (short for "examples") that behaves as follows: *EX* first draws a point $x \in X$ randomly according to the distribution D. Then with probability $c(x)$, *EX* returns the labeled example $(x, 1)$, and with probability $1 - c(x)$ it returns $(x, 0)$. Thus, $c(x)$ is the conditional probability that example x is labeled 1. Further note that the learning algorithm never has direct access to these conditional probabilities $c(x)$, but only to random examples whose labels are distributed according to these unknown probabilities.

Let h be a function mapping X into $\{0, 1\}$; we call such a function a *decision rule*. We define the *predictive error* of h on c with respect to D, denoted $R_D(c, h)$, as the probability that h will misclassify a randomly drawn point from *EX*. If h minimizes $R_D(c, \cdot)$, then we say that h is a *best decision rule*, or a *Bayes optimal decision rule*, for c. We say that h is an *ϵ-good* decision rule for c if $R_D(c, h) \leq R_D(c, \tilde{h}) + \epsilon$, where $\tilde{h}$ is a best decision rule. Thus we ask that h be nearly as good as the best decision rule for c.

The *projection* of the p-concept c is the function $\pi_c : X \to \{0, 1\}$ that is 1 if $c(x) \geq 1/2$ and 0 if $c(x) < 1/2$. It is well-known and easy to show that for any target p-concept c, its projection π_c is a Bayes optimal decision rule.

In this paper we are primarily interested not in the problem of finding a good decision rule, but in that of producing an accurate real-valued approximation to the target p-concept itself. Thus, we wish to infer a good *model of probability* with respect to the target distribution. We say that a p-concept h is an *(ϵ, γ)-good model of probability* of c with respect to D if we have $\mathbf{Pr}_{x \in D}[|h(x) - c(x)| > \gamma] \leq \epsilon$. Thus, the value of h must be near that of c on most points x.

We are now ready to describe our model for learning p-concepts. Let $\mathcal{C}$ be a p-concept class over domain X. We say that $\mathcal{C}$ is *learnable with a model of probability* (respectively, *learnable with a decision rule*) if there is an algorithm A such that for any target p-concept $c \in \mathcal{C}$, for any target distribution D over X, for any inputs $\epsilon > 0$, $\delta > 0$, and $\gamma > 0$, algorithm A, given access to *EX*, halts and with probability at least $1 - \delta$ outputs a p-concept h that is an (ϵ, γ)-good model of probability (respectively, an ϵ-good decision rule) for c with respect to D. Note that this model of learning p-concepts generalizes Valiant's model for learning deterministic concepts.

We say that $\mathcal{C}$ is *polynomially learnable* (with either a model of probability or a decision rule) if there is a learning algorithm A that runs in time polynomial in $1/\epsilon$, $1/\delta$ and, where appropriate, $1/\gamma$. (In fact, an equivalent formulation is obtained by requiring the running time to be polynomial in $\log(1/\delta)$, rather than $1/\delta$. See Section 4.) Often the p-concept class $\mathcal{C}$ will be parameterized by a complexity parameter n, that is $\mathcal{C} = \bigcup_{n \geq 1} \mathcal{C}_n$, and all p-concepts in $\mathcal{C}_n$ share a common subdomain X_n, and $X = \bigcup_{n \geq 1} X_n$. In such cases we also allow a polynomial dependence on n.

Our first lemma shows that a good model of probability can always be efficiently used as a good decision rule; thus, learning with a model of probability is a harder problem than decision-rule learning.

Lemma 2.1 *Let $\mathcal{C}$ be a class of p-concepts. If $\mathcal{C}$ is (polynomially) learnable with a model of probability, then $\mathcal{C}$ is (polynomially) learnable with a decision rule.*

Proof: To prove the lemma, we show that the projection of a good model of probability can be used as a good decision rule. In particular, we show that if h is an (ϵ, γ)-good model of probability, then π_h is an $(\epsilon + 2\gamma)$-good decision rule. Thus, by choosing ϵ and γ appropriately, an arbitrarily good decision rule can be found by the assumed algorithm for learning with a model of probability.

Let $x \in X$, and suppose $|h(x) - c(x)| \leq \gamma$. If $|c(x) - 1/2| > \gamma$, then clearly $\pi_h(x) = \pi_c(x)$. On the other hand, if $|c(x) - 1/2| \leq \gamma$, then it may be that $\pi_h(x) \neq \pi_c(x)$. However, the chance that $\pi_c(x)$ agrees with a random label for x (chosen according to c) is at most $1/2 + \gamma$, while the chance that $\pi_h(x)$ agrees with the random label is at least $1/2 - \gamma$.

Thus, the difference in predictive error between π_c and π_h (taken over a random choice of an instance and its label) is at most $\epsilon + (1 - \epsilon) \cdot 2\gamma \leq \epsilon + 2\gamma$. ■

2.1 Alternative formulations

In addition to the formulation given above, there are various other natural ways of expressing the fact that some hypothesis p-concept h is "close" to the target c. For example, we might say that h is a good model of probability for c if the average difference between the two functions is small, i.e., if the *variational distance* $\mathbf{E}_{x \in D}[|h(x) - c(x)|]$ is small. Alternatively, our goal might be to make small the *quadratic distance* (i.e., the expected square of the difference between the functions).

As we will see in the following sections, these alternative definitions are sometimes easier to work with than the "official" definition given above. The next lemma shows that the three formulations are equivalent modulo polynomial-time computation.

Lemma 2.2 *Let h and c be p-concepts, and let D be a target distribution on domain X. Let $e_1 = \mathbf{E}_{x \in D}[|h(x) - c(x)|]$ and let $e_2 = \mathbf{E}_{x \in D}\left[(h(x) - c(x))^2\right]$. Then*

- *$e_2 \leq e_1 \leq \sqrt{e_2}$;*
- *for any $\gamma > 0$, h is both an $(e_1/\gamma, \gamma)$- and an $(e_2/\gamma^2, \gamma)$-good model of probability for c;*
- *if h is an (ϵ, γ)-good model of probability, then $e_1 \leq \epsilon + \gamma$, and $e_2 \leq \epsilon + \gamma^2$.*

Proof: Since $|h(x) - c(x)| \leq 1$ for all x, it is clear that $e_2 \leq e_1$. Also, it is well known that $(\mathbf{E}[Y])^2 \leq \mathbf{E}[Y^2]$ for any random variable Y. Thus, $e_1 \leq \sqrt{e_2}$.

Let $\gamma > 0$. Then by Markov's inequality,

$$\mathbf{Pr}_{x \in D}[|h(x) - c(x)| > \gamma] \leq e_1/\gamma.$$

Similarly,

$$\mathbf{Pr}_{x\in D}\left[|h(x)-c(x)|>\gamma\right] = \mathbf{Pr}_{x\in D}\left[(h(x)-c(x))^2>\gamma^2\right] \le e_2/\gamma^2.$$

These imply the second part of the lemma.

Finally, suppose h is an (ϵ,γ)-good model of probability. Then

$$e_1 \le \mathbf{Pr}_{x\in D}\left[|h(x)-c(x)|>\gamma\right]\cdot 1 + \mathbf{Pr}_{x\in D}\left[|h(x)-c(x)|\le\gamma\right]\cdot\gamma \le \epsilon + (1-\epsilon)\gamma \le \epsilon+\gamma.$$

Similarly, $e_2 \le \epsilon + \gamma^2$. ■

Theorem 2.3 *Let $\mathcal{C}$ be a class of p-concepts, let c be the target p-concept, and let D be a target distribution on domain X. Then, assuming access to oracle EX, the following computational problems are equivalent (i.e., if one is solvable, then so are the others):*

1. *finding, with probability at least $1-\delta$, an (ϵ,γ)-good model of probability in time polynomial in $1/\epsilon$, $1/\delta$ and $1/\gamma$;*
2. *finding, with probability at least $1-\delta$, a hypothesis h such that $\mathbf{E}_{x\in D}\left[|h(x)-c(x)|\right]\le\epsilon$ in time polynomial in $1/\epsilon$ and $1/\delta$;*
3. *finding, with probability at least $1-\delta$, a hypothesis h such that $\mathbf{E}_{x\in D}\left[(h(x)-c(x))^2\right]\le\epsilon$ in time polynomial in $1/\epsilon$ and $1/\delta$.*

Proof: Lemma 2.2 implies the following:

- Given an algorithm A_1 for solving problem 1, problem 2 can be solved by running A_1 with ϵ and γ each set to $\epsilon/2$.
- Given an algorithm A_2 for solving problem 2, problem 3 can be solved by running A_2 directly.
- Given an algorithm A_3 for solving problem 3, problem 1 can be solved by running A_3 with ϵ set to $\epsilon\gamma^2$.

■

In addition to the formulations given by Theorem 2.3, Yamanishi [31] shows that an equivalent problem is to find, in polynomial time with high probability and for given ϵ, a hypothesis h such that $\mathbf{E}_{x\in D}\left[(\sqrt{h(x)}-\sqrt{c(x)})^2\right]\le\epsilon$. (This quantity is known as the *Hellinger distance.*) Finally, Abe, Takeuchi and Warmuth [1] have shown that all of these problems are equivalent (modulo polynomial-time computation) to the problem of finding, with high probability and for given ϵ, a hypothesis with small *Kullback-Liebler divergence*, i.e., a hypothesis h for which

$$\mathbf{E}_{x\in D}\left[c(x)\lg\left(\frac{c(x)}{h(x)}\right)+(1-c(x))\lg\left(\frac{1-c(x)}{1-h(x)}\right)\right]\le\epsilon.$$

2.2 Chernoff bounds

Several times, in later sections of this paper, we will make use of the following bounds on the tails of a binomial distribution [4, 13].

Lemma 2.4 (Chernoff Bounds) *Let $X_1, \ldots, X_m$ be a sequence of m independent Bernoulli trials, each succeeding with probability p so that $\mathbf{E}[X_i] = p$. Let $S = X_1 + \cdots + X_m$ be the random variable describing the total number of successes. Then for $0 \leq \gamma \leq 1$, the following hold:*

- *(additive form)* $\mathbf{Pr}[S > (p+\gamma)m] \leq e^{-2m\gamma^2}$, *and* $\mathbf{Pr}[S < (p-\gamma)m] \leq e^{-2m\gamma^2}$;
- *(multiplicative form)* $\mathbf{Pr}[S > (1+\gamma)pm] \leq e^{-\gamma^2 mp/3}$, *and* $\mathbf{Pr}[S < (1-\gamma)pm] \leq e^{-\gamma^2 mp/2}$.

The additive form (also known as Hoeffding's inequality) holds also if $X_1, \ldots, X_m$ are independent identically distributed random variables with range in $[0,1]$.

3 Efficient algorithms: The direct approach

In this section, we describe efficient algorithms for learning good models of probability based on first principles and proved correct by direct arguments. Later arguments will rely on an underlying theory of p-concept learning that is developed in subsequent sections. We begin with a p-concept class motivated by the problem of modeling "tallness" discussed in the introduction.

3.1 Increasing functions

Theorem 3.1 *The p-concept class of all nondecreasing functions $c : \mathbb{R} \to [0,1]$ is polynomially learnable with a model of probability.*

Proof: We prove the result in slightly greater generality for any domain X linearly ordered by some ordering "$\leq$." Given positive ϵ, δ and γ, let $t = \lceil 4/\epsilon\gamma \rceil$, and let

$$s = \left\lceil \max\left\{ \frac{64\ln(2^{21}/(\epsilon\gamma)^2\delta)}{\epsilon\gamma}, \frac{2\ln(4t/\delta)}{\gamma^2} \right\} \right\rceil.$$

Our algorithm begins by drawing a labeled sample of $m = st$ examples (x_i, b_i). The examples are sorted and reindexed so that $x_1 \leq \cdots \leq x_m$. In fact, we assume initially that no instance occurs twice in the sample so that $x_1 < \cdots < x_m$. Later, we show how this assumption can be removed.

The set X can naturally be partitioned into t disjoint intervals I_j, each containing exactly s instances of the sample; specifically, we let $I_1 = (-\infty, x_s]$; $I_j = (x_{(j-1)s}, x_{js}]$ for $j = 2, 3, \ldots, t-1$; and $I_t = (x_{(t-1)s}, \infty)$. For $1 \leq j \leq t$, let $\hat{p}_j = (1/s) \cdot \sum_{x_i \in I_j} b_i$. Thus, $\hat{p}_j$ is an estimate of the probability p_j that a random instance in I_j is labeled 1. Our algorithm outputs a step function h defined in a natural manner: for $x \in I_j$, we define $h(x) = \hat{p}_j$.

This algorithm clearly runs in polynomial time. We argue next that the output hypothesis h is an (ϵ, γ)-good model of probability (with high probability). Here are the high-level ideas: first, we show that (with high probability) each interval has weight approximately $\epsilon\gamma$ under the target distribution. Next we show that if c increases by roughly γ or less on the interval I_j, then h is close to c on all points in the interval. On the other hand, since c is nondecreasing and bounded between 0 and 1, c can increase by more than γ in at most $1/\gamma$ intervals; since these "bad" intervals have total weight at most ϵ, h is a good model of probability.

Specifically, we can apply the uniform convergence results of Vapnik and Chervonenkis [29] to show that, with high probability, each interval I_j has probability at most $\epsilon\gamma/2$. Let S be the set of all intervals on X. Then Theorem 2 of their paper shows that, with probability at least $1-\delta/2$, for the sample size m chosen by our algorithm, the relative fraction of points of the sample occurring in *any* interval of S is within $\epsilon\gamma/4$ of the true weight of the interval under the target distribution. In particular, since each interval I_j contains $1/t \leq \epsilon\gamma/4$ of the instances in the sample, the weight of I_j under the target distribution is at most $\epsilon\gamma/2$. (Technically, their results rely on certain measurability assumptions which may depend on the choice of X. However, these assumptions are satisfied when $X = \mathbb{R}$.)

Let $q_j = c(x_{js})$ for $1 \leq j < t$, and let $q_0 = 0$ and $q_t = 1$. Then for $x \in I_j$, it is clear that $q_{j-1} \leq c(x) \leq q_j$ since c is nondecreasing. In particular, this is true for each $x_i \in I_j$. Thus, each point $x_i \in I_j$ is labeled 1 with probability $c(x_i) \geq q_{j-1}$, and so, for each j, $\hat{p}_j \geq q_{j-1} - \gamma/2$ with probability at least $1-\delta/4t$; this follows from the fact that $s \geq (2/\gamma^2)\cdot\ln(4t/\delta)$, and by applying the additive form of Chernoff bounds given in Lemma 2.4. Similarly, $\hat{p}_j \leq q_j + \gamma/2$ with probability at least $1-\delta/4t$. Thus, it follows that $q_{j-1} - \gamma/2 \leq \hat{p}_j \leq q_j + \gamma/2$ for all j with probability at least $1-\delta/2$. Hence, if $q_j - q_{j-1} \leq \gamma/2$, then $|h(x) - c(x)| \leq \gamma$ for $x \in I_j$.

On the other hand, $q_j - q_{j-1}$ can exceed $\gamma/2$ for at most $2/\gamma$ values of j since c is nondecreasing, and bounded between 0 and 1. Since each of these "bad" intervals has probability weight at most $\epsilon\gamma/2$, the sum total probability of these intervals under D is at most ϵ. Thus, h is an (ϵ, γ)-good model of probability.

Finally, we show how to insure that the sample does not contain the same instance more than once. Such a situation could be problematic for our algorithm since it might cause some of the intervals defined above to be empty, or to contain too many sample points.

The idea is to replace the given domain X and target distribution D with a new domain X' and distribution D' under which the same instance is very unlikely to occur twice. In particular, we let $X' = X \times T$ and $D' = D \times U$ where U is the uniform distribution on the set $T = \{0, \ldots, 2^k - 1\}$, and $k = \lceil 2\lg m + \lg(1/\delta)\rceil$. Then X' is linearly ordered under the lexicographic ordering (i.e., $(x, r) \leq (y, s)$ if and only if $x < y$, or $x = y$ and $r \leq s$). Also, the chance that any pair of instances are the same in a sample of size m drawn according to D' is at most $\binom{m}{2}\cdot 2^{-k} \leq m^2 \cdot 2^{-k-1} \leq \delta/2$.

In addition, given a random source of instances from X drawn according to D, we can easily simulate the random choice of instances from X' according to D': given $x \in X$, we simply draw a random number r uniformly from T, yielding an instance (x, r) with distribution D' (x's label is not altered). Thus, the previously described algorithm can be simulated (with δ replaced by $\delta/2$) on domain X'. If the same instance occurs twice in the sample, the algorithm simply fails — as argued above, this will happen with probability at most $\delta/2$. Thus, with probability at least $1-\delta$, the algorithm returns an (ϵ, γ)-good hypothesis h (with respect to X'). This hypothesis can be used to estimate $c(x)$ for a given point $x \in X$ by randomly choosing $r \in T$ and evaluating $h((x, r))$. Although this yields a randomized hypothesis h', it remains true that the probability (over choices of $x \in X$ and the randomization of h') that h' differs by more than γ from c is at most ϵ. Thus, h' is an (ϵ, γ)-good model of probability if h is. ■

This algorithm can be modified to learn with a model of probability any function over the real line with at most d extremal points; the running time is then polynomial in d, $1/\epsilon$, $\log(1/\delta)$ and $1/\gamma$.

In principle, the algorithm of Theorem 3.1 could be used to learn the p-concept class of non-decreasing functions with a decision rule (by applying Lemma 2.1). However, a much simpler and more efficient algorithm exists that we give in Section 5.

3.2 Probabilistic decision lists

We turn next to the problem of learning a probabilistic analog of Rivest's decision lists [24]. We define such lists with respect to a basis $\mathcal{F}_n$ of Boolean-valued functions on the domain $\{0,1\}^n$. We assume always that $\mathcal{F}_n$ contains the constant function 1. Then a *probabilistic decision list c over basis $\mathcal{F}_n$* is given by a list $(f_1, r_1), \ldots, (f_s, r_s)$, where each $f_i \in \mathcal{F}_n$, and each $r_i \in [0, 1]$. We also assume that f_s is the constant function 1. For any assignment x in the domain, $c(x)$ is defined to be r_j, where j is the least index for which $f_j(x) = 1$. In other words, the functions in $\mathcal{F}_n$ are tested one by one in the order specified by the list, until a function which evaluates to 1 on x is encountered; the corresponding real number r_j is then the probability that x is labeled 1.

Rivest does not define decision lists with respect to a general basis as is done here. Rather, in his definition, a decision list only tests the values of monomials. That is, he defines decision lists specifically with respect to the basis consisting of all conjunctions of literals. He goes on to define the class k-DL of decision lists in which each monomial occurring in the list is a conjunction of k or fewer literals. Thus, this class is over the basis of all monomials of size at most k. Rivest describes an efficient algorithm for learning the class k-DL, when k is any fixed constant.

Below, we describe an efficient algorithm for learning a special class of probabilistic decision lists over any basis $\mathcal{F}_n$. The running time of this algorithm is polynomial in all of the usual parameters, in addition to $|\mathcal{F}_n|$, and the maximum time needed to evaluate any function f in $\mathcal{F}_n$. Thus, in particular, this implies a polynomial-time algorithm for the same basis considered by Rivest, namely, the set of all conjunctions of k or fewer literals, for k a fixed constant.

Let c be a probabilistic decision list over basis $\mathcal{F}_n$, given by the list $(f_1, r_1), \ldots, (f_s, r_s)$. For $\omega \in [0, 1]$, we say that c is a probabilistic decision list *with ω-converging probabilities* if $|r_i - \omega| \geq |r_{i+1} - \omega|$ for $1 \leq i < s$. Below, we describe an algorithm for inferring such lists when ω is known. As a special case, when $\omega = 0$, this algorithm can be used to learn probabilistic decision lists *with decreasing probabilities*, i.e., lists in which $r_i \geq r_j$ for $i \leq j$.

Perhaps the most natural case occurs when $\omega = 1/2$. In this case, we say that c is a probabilistic decision list *with decreasing certainty* since instances with the most certain outcomes (labels) are handled at the beginning of the list. For instance, a college's admissions process (see Section 1) might be naturally modeled in this manner as a list of criteria for determining admission, ordered by importance: for example, if the student has straight A's, then he should be admitted with 90% probability; otherwise, if he did poorly on his SAT's, then he should be rejected with 85% probability; otherwise, if he was class president, then he should be accepted with 75% probability; and so on. Note that the class of probabilistic decision lists with decreasing certainty includes the class of ordinary (deterministic) decision lists over the same basis.

We also note that the algorithm given below in Theorem 3.2 can be applied to learn ordinary decision lists when the supplied examples are "noisy." Specifically, consider the problem of learning a deterministic decision list c given by the list $(f_1, b_1), \ldots, (f_s, b_s)$ where each f_i is in the basis $\mathcal{F}_n$, and, since the list is deterministic, each $b_i \in \{0, 1\}$. Suppose further that the label of each example is flipped (i.e., reversed) randomly with probability $\eta < 1/2$. This random misclassification noise model is considered, for instance, by Angluin and Laird [3]. Note that the observed behavior in such a situation can be modeled naturally by the probabilistic decision list c' given by $(f_1, |b_1 - \eta|), \ldots, (f_s, |b_s - \eta|)$. That is, $c'(x)$ is the probability that x is labeled 1 by a noisy oracle for c. Clearly, c' is a probabilistic decision list with 1/2-converging probabilities. Thus, we can apply the efficient learning algorithm for this class (described below) to obtain a good model of probability h for c'. If we choose $\gamma < 1/2 - \eta$, then it can be seen that the projection of h is a good approximation of c; that is, with probability at least $1 - \delta$, a hypothesis h is obtained for which $\mathbf{Pr}_{x \in D}[\pi_h(x) \neq c(x)] \leq \epsilon$. (Technically, this algorithm assumes that η, or an upper bound on η, is known. However, if no such bound is known, Angluin and Laird [3] give a technique for finding a good bound using a kind of "binary search.")

Thus, a corollary of Theorem 3.2 is a proof that deterministic decision lists are efficiently learnable even when the supplied examples are randomly misclassified with probability η. The running time is then polynomial in $1/(1 - 2\eta)$, in addition to the usual other parameters. This specifically answers an open question proposed by Rivest [24] concerning the learnability of decision lists in such a noisy setting. (This problem of learning noisy decision lists was solved independently by Sakakibara [25].)

Theorem 3.2 *Let $\omega \in [0, 1]$ be fixed, and let $\mathcal{F}_n$ be a basis of functions. Then the p-concept class of probabilistic decision lists over basis $\mathcal{F}_n$ with ω-converging probabilities is learnable with a model of probability (assuming both ω and $\mathcal{F}_n$ are known). Specifically, this class can be learned in time polynomial in $1/\epsilon$, $1/\gamma$, $\log(1/\delta)$, n, $|\mathcal{F}_n|$ and the maximum time needed to evaluate any function in $\mathcal{F}_n$.*

Proof: Our learning algorithm for this p-concept class is shown in Figure 1. As usual, the algorithm begins by drawing a large sample S of size m which will be used to construct a hypothesis probabilistic decision list L. (Note that S and all subsets derived from S are *multisets* — they are "sets" which may contain multiple copies of the same example.)

Assume for convenience that the functions in $\mathcal{F}_n$ are indexed so that the target p-concept c is given by the list $(f_1, r_1), \ldots, (f_s, r_s)$. (Of course, the learning algorithm is not aware of this.) We

Input: $\omega \in [0,1]$
basis $\mathcal{F}_n = \{f_1, \ldots, f_s\}$
$\epsilon,\ \delta,\ \gamma > 0$
access to random examples of a probabilistic decision list over basis $\mathcal{F}_n$ with ω-converging probabilities

Output: with probability at least $1 - \delta$, an (ϵ, γ)-good model of probability

Procedure:

$L \leftarrow$ empty list
$J \leftarrow \{1, \ldots, s\}$
obtain a sample S of $m = \lceil (32s/\epsilon^3\gamma^2) \cdot \ln(2^{s+2}s/\delta) \rceil$ random examples
repeat
  if $|\{(x,b) \in S : f_j(x) = 1\}| \leq m\epsilon/4s$ for some $j \in J$ **then**
    $t \leftarrow j$
    $\hat{p}_t \leftarrow 0$
  else
    for $j \in J$: $\hat{p}_j \leftarrow |\{(x,b) \in S : f_j(x) = 1 \wedge b = 1\}| \div |\{(x,b) \in S : f_j(x) = 1\}|$
    choose t that maximizes $|\hat{p}_j - \omega|$
  $L \leftarrow L, (f_t, \hat{p}_t)$
  $S \leftarrow \{(x,b) \in S : f_t(x) = 0\}$
  $J \leftarrow J - \{t\}$
until $J = \emptyset$
output L

Figure 1: An algorithm for learning probabilistic decision lists with ω-converging probabilities.

also assume without loss of generality that every function in the basis $\mathcal{F}_n$ occurs in the target list so that $s = |\mathcal{F}_n|$.

Here is the intuition behind our algorithm: using the sample, we might estimate the probability p_i that a positive random example $(x, 1)$ is drawn, given that $f_i(x) = 1$. It can be shown to follow from the definition of ω-converging decision lists that $|p_1 - \omega| \geq |p_i - \omega|$ for all i. This suggests a technique for identifying the first variable in the list: if our estimates $\hat{p}_i$ are sufficiently accurate, we would expect $|\hat{p}_i - \omega|$ to be maximized when $i = 1$. This is the approach taken by our algorithm: the function f_i for which $|\hat{p}_i - \omega|$ is greatest is placed at the head of the hypothesis list. The remainder of the list is constructed iteratively using the part of the sample on which $f_i(x) = 0$.

For $I \subset \{1, \ldots, s\}$ and $j \in \{1, \ldots, s\}$, let $A(I, j)$ be the set of all instances x for which $f_j(x) = 1$ and $f_i(x) = 0$ for all $i \in I$. Let

$$u(I, j) = \mathbf{Pr}_{x \in D}\left[x \in A(I, j)\right]$$

and

$$v(I, j) = \mathbf{Pr}_{(x,b) \in EX}\left[b = 1 \mid x \in A(I, j)\right].$$

Also, let $\hat{u}(I,j)$ and $\hat{v}(I,j)$ be empirical estimates of these quantities derivable from the sample S in the obvious manner.

Let $I \subset \{1,\ldots,s\}$ and $j \in \{1,\ldots,s\}$ be fixed. Then, using the multiplicative form of Chernoff bounds given by Lemma 2.4, it follows that if $u(I,j) > \epsilon/2s$ then, since $m \geq (16s/\epsilon)\cdot\ln(2^{s+1}s/\delta)$,

$$\hat{u}(I,j) \geq \tfrac{1}{2}\cdot u(I,j)$$

with probability at least $1-\delta/(s\cdot 2^{s+1})$. Furthermore, if $\hat{u}(I,j) > \epsilon/4s$, then the number of instances $x \in A(I,j)$ included in S is at least $m\epsilon/4s \geq (8/\epsilon^2\gamma^2)\cdot\ln(2^{s+2}s/\delta)$. Thus, applying the additive form of Chernoff bounds, we see that

$$|v(I,j) - \hat{v}(I,j)| \leq \epsilon\gamma/4$$

with probability at least $1-\delta/2^{s+1}s$, assuming $\hat{u}(I,j) > \epsilon/4s$.

Thus, with probability at least $1-\delta$, a sample S is chosen such that for all $I \subset \{1,\ldots,s\}$ and for all $j \in \{1,\ldots,s\}$, we have that

$$u(I,j) \leq \max\left(\frac{\epsilon}{2s}, 2\hat{u}(I,j)\right), \tag{1}$$

and, whenever $\hat{u}(I,j) > \epsilon/4s$, we also have that

$$|v(I,j) - \hat{v}(I,j)| \leq \frac{\epsilon\gamma}{4}. \tag{2}$$

We assume henceforth that all of the empirical estimates $\hat{u}(I,j)$ and $\hat{v}(I,j)$ satisfy the conditions described above. As just argued, this will be the case with probability at least $1-\delta$. To complete the proof, we show that this assumption implies that the algorithm's output hypothesis h is an (ϵ,γ)-good model of probability.

Suppose h is given by the list $(f_{t_1}, r'_1),\ldots,(f_{t_s}, r'_s)$. Let $T_i = \{t_1,\ldots,t_i\}$. To prove that h is an (ϵ,γ)-good model of probability, we show that, for $1 \leq i \leq s$, either

$$\mathbf{Pr}_{x\in D}\left[x \in A(T_{i-1}, t_i)\right] \leq \epsilon/2s \tag{3}$$

or

$$\mathbf{Pr}_{x\in D}\left[|h(x) - c(x)| > \gamma \mid x \in A(T_{i-1}, t_i)\right] \leq \epsilon/2. \tag{4}$$

Note that the sets $A(T_{i-1}, t_i)$ are disjoint. Thus, this implies

$$\begin{aligned}
&\mathbf{Pr}_{x\in D}\left[|h(x) - c(x)| > \gamma\right] \\
&= \sum_{i=1}^{s} \mathbf{Pr}_{x\in D}\left[|h(x) - c(x)| > \gamma \mid x \in A(T_{i-1}, t_i)\right]\cdot \mathbf{Pr}_{x\in D}\left[x \in A(T_{i-1}, t_i)\right] \\
&\leq \epsilon
\end{aligned}$$

as can be seen by breaking the sum into two parts based on whether $\mathbf{Pr}_{x\in D}\left[x \in A(T_{i-1}, t_i)\right]$ exceeds or does not exceed $\epsilon/2s$.

Fix i, and consider the ith iteration of our algorithm. Prior to the extension of L at line 11, the hypothesis list is $(f_{t_1}, r'_1), \ldots, (f_{t_{i-1}}, r'_{i-1})$. Let $C_j = A(T_{i-1}, j)$. Also, let $p_j = v(T_{i-1}, j)$, and observe that, as defined in the figure, $\hat{p}_j = \hat{v}(T_{i-1}, j)$. This follows from the fact that, at this point in the execution of the algorithm, all examples (x, b) in S are such that $f_k(x) = 0$ for $k \in T_{i-1}$.

Let t be as in the figure (i.e., $t = t_i$). If t was chosen at line 6, then $\hat{u}(T_{i-1}, t) \leq \epsilon/4s$, and so $u(T_{i-1}, t) \leq \epsilon/2s$ by equation (1). Thus, in this case, equation (3) holds by definition of $u(I, j)$.

Otherwise, for all $j \in J$, $\hat{u}(T_{i-1}, j) > \epsilon/4s$, and thus, $|p_j - \hat{p}_j| \leq \epsilon\gamma/4$ by equation (2). We wish to prove that equation (4) holds in this case, i.e., that

$$\mathbf{Pr}_{x\in D}\left[|\hat{p}_t - c(x)| > \gamma \mid x \in C_t\right] \leq \epsilon/2.$$

Let u be the smallest member of J. Then $p_u = r_u$ by definition of decision lists. Also, since c is given by a list with ω-converging probabilities, $|r_u - \omega| \geq |r_j - \omega|$ for $j \geq u$. Thus, by our choice of t, for $j \in J$,

$$|r_j - \omega| \leq |r_u - \omega| = |p_u - \omega| \leq |\hat{p}_u - \omega| + \epsilon\gamma/4 \leq |\hat{p}_t - \omega| + \epsilon\gamma/4.$$

Suppose $\hat{p}_t \geq \omega$. Then clearly $r_j \leq \hat{p}_t + \epsilon\gamma/4$ for $j \in J$, and thus $c(x) \leq \hat{p}_t + \epsilon\gamma/4 \leq \hat{p}_t + \gamma$ whenever $x \in C_t$. Let z be the probability that an x is chosen for which $c(x) < \hat{p}_t - \gamma$, given that x is in C_t:

$$z = \mathbf{Pr}_{x\in D}\left[c(x) < \hat{p}_t - \gamma \mid x \in C_t\right].$$

Then

$$\begin{aligned}
p_t &= \mathbf{E}_{x\in D}\left[c(x) \mid x \in C_t\right] \\
&\leq z(\hat{p}_t - \gamma) + (1 - z)(\hat{p}_t + \epsilon\gamma/4) \\
&\leq z(p_t + \epsilon\gamma/4 - \gamma) + (1 - z)(p_t + \epsilon\gamma/2) \\
&\leq p_t + \epsilon\gamma/2 - \gamma z.
\end{aligned}$$

This implies $z \leq \epsilon/2$, and so (4) holds in this case. The proof of (4) is symmetric when $\hat{p}_t \leq \omega$.

The algorithm of Figure 1 clearly runs in polynomial time. ■

It is an open question whether this class is learnable when ω is unknown.

The class of probabilistic decision lists has also been considered by Yamanishi [31]. He describes an algorithm, based on the principle of minimum description length, for learning a model of probability for p-concepts in this class; however, his algorithm is *not* computationally efficient. Also, Aiello and Mihail [2] have recently described an efficient algorithm for learning arbitrary probabilistic decision lists over the basis consisting of all literals in the special case that D is the uniform distribution.

3.3 Hidden-variable problems

We next consider p-concept classes motivated by *hidden-variable* problems, in which there is an underlying deterministic concept, but the settings of some of the relevant variables are invisible to the learning algorithm, resulting in apparent probabilistic behavior. A *visible monomial* p-concept is defined over $\{0,1\}^n$ by a pair (M, α), where M is a monomial over the *visible* Boolean variables $x_1, \ldots, x_n$ and $\alpha \in [0,1]$. The associated p-concept c is defined for $x \in \{0,1\}^n$ to be $c(x) = \alpha \cdot M(x)$. We conceptually regard the true deterministic concept as having the form $M \wedge I$, where I is a deterministic concept over the *hidden variables*. We interpret α as the probability that the settings of the invisible variables satisfy I. Note that we assume independence between the settings for the variables of M and those for I.

For instance, I might itself be a monomial, in which case the underlying target concept is a conjunction of literals, some which are visible and some which are hidden.

Visible monomials model well situations in which certain observable conditions are requisite to some outcome, but in which these conditions are not in themselves enough to determine the outcome with certainty. Thus, the conditions are necessary, but not sufficient, and, when the conditions are met, the final outcome may be uncertain. For instance, if you are handed a drink that is brown and fizzes and tastes sweet, then the drink might be Coke; on the other hand, it might not be Coke (it could be Pepsi). In any case, if the drink lacks any one of these qualities, then it certainly cannot be "the real thing."

We note that the algorithm described in the proof below can be easily extended to learn any p-concept c of the form $c = \alpha c_0$ where α is an unknown constant in $[0,1]$ and c_0 is a deterministic concept from some known concept class for which there exists an efficient algorithm that, like the algorithm V described in the proof, requires positive examples only, and outputs hypotheses with one-sided error on the positive-examples distribution only. For instance, Valiant [27] describes such an algorithm for learning k-CNF (the class of Boolean formulas consisting of a conjunction of clauses, each a disjunction of at most k literals).

Theorem 3.3 *The class of visible monomial p-concepts is polynomially learnable with a model of probability.*

Proof: Let the target p-concept c be defined by the pair (M, α), and let the target distribution over $\{0,1\}^n$ be D. We describe an algorithm that, given $\epsilon, \delta > 0$, outputs with probability at least $1-\delta$ a hypothesis h for which $\mathbf{E}_{x\in D}\left[|h(x) - c(x)|\right] \le \epsilon$; Theorem 2.3 implies that such an algorithm can be converted into one that learns a good model of probability.

The first step of the learning algorithm is to obtain an estimate $\hat{p}$ of $p = \mathbf{Pr}_{(x,b)\in EX}\left[b = 1\right]$ that, with probability at least $1-\delta/3$, is such that $|p - \hat{p}| \le \epsilon/3$. If $\hat{p} \le 2\epsilon/3$, then the algorithm outputs the hypothesis $h(x) \equiv 0$. Assuming $\hat{p}$ has the desired accuracy, we have $\mathbf{E}_{x\in D}\left[|c(x) - h(x)|\right] \le \epsilon$ in this case as desired, since $p = \mathbf{E}_{x\in D}\left[c(x)\right] \le \epsilon$. Otherwise, $\hat{p} > 2\epsilon/3$, and we can assume henceforth that $p \ge \epsilon/3$ (as is the case with probability at least $1 - \delta/3$).

Next our algorithm attempts to learn a good approximation of M. This is done using Valiant's algorithm [27], here denoted V, for learning monomials from positive examples only in the distribution-free deterministic model. Algorithm V, which we here use as a "black-box" subroutine, has the following properties: the algorithm takes as input positive ϵ and δ, and a source of positive examples of some monomial M, each chosen randomly according to some fixed, arbitrary distribution D^+ on the set of all positive examples. After running for time polynomial in $1/\epsilon$, $\log(1/\delta)$ and n, V outputs a monomial $\hat{M}$ that, with high probability, has error at most ϵ for the positive examples of M, and has zero error for the negative examples. That is, with probability at least $1-\delta$, $\hat{M}$ is such that:

$$\mathbf{Pr}_{x\in D^+}\left[\hat{M}(x)=0\right] \le \epsilon,$$

and also

$$M(x)=0 \Rightarrow \hat{M}(x)=0.$$

Our algorithm simulates V with V's parameter ϵ set to $\epsilon/4$, and δ set to $\delta/3$. We provide V with a simulated oracle EX' which supplies V with only positively labeled examples. Specifically, when V requests an example, EX' draws examples from EX until an example $(x,1)$ is received; this instance x is then provided to V.

Note that if x is labeled positively by EX, then $c(x)>0$ and so $M(x)=1$. Thus, V is only supplied with positive examples. Note also that the probability of drawing a positively labeled example from EX equals p. Since $p \ge \epsilon/3$, it follows that the expected running time of EX' is at most $O(1/\epsilon)$.

The probability that EX' outputs some instance x is just

$$\begin{aligned} D^+(x) &= \mathbf{Pr}_{(y,b)\in EX}\left[y=x \mid b=1\right] \\ &= \frac{\mathbf{Pr}_{(y,b)\in EX}\left[y=x \wedge b=1\right]}{\mathbf{Pr}_{(y,b)\in EX}\left[b=1\right]} \\ &= \frac{\alpha M(x)\cdot D(x)}{\alpha\cdot \mathbf{Pr}_{y\in D}\left[M(y)=1\right]} \\ &= \frac{M(x)D(x)}{\mathbf{Pr}_{y\in D}\left[M(y)=1\right]} \\ &= \mathbf{Pr}_{y\in D}\left[y=x \mid M(y)=1\right]. \end{aligned}$$

With probability at least $1-\delta/3$, V outputs a hypothesis $\hat{M}$ which is such that $\hat{M}(x)=0$ whenever $M(x)=0$, and

$$\mathbf{Pr}_{x\in D^+}\left[\hat{M}(x)=0\right] \le \epsilon/4.$$

Our algorithm next obtains an estimate $\hat{\alpha}$ of $\alpha' = \mathbf{Pr}_{(x,b)\in EX}\left[b=1 \mid \hat{M}(x)=1\right]$ that, with probability at least $1-\delta/3$, is such that $|\alpha'-\hat{\alpha}| \le \epsilon/2$. Such an estimate can be derived from a polynomial-size sample since

$$\mathbf{Pr}_{x\in D}\left[\hat{M}(x)=1\right] \ge (1-\epsilon/4)\cdot \mathbf{Pr}_{x\in D}\left[M(x)=1\right] \ge (1-\epsilon/4)p \ge (1-\epsilon/4)(\epsilon/3).$$

The algorithm outputs the hypothesis h defined by $(\hat{M},\hat{\alpha})$; we argue next that h is, with probability at least $1-\delta$, within ϵ of c on average.

As noted above, $\hat{M}$ has the property that

$$\mathbf{Pr}_{x\in D}\left[\hat{M}(x)=0 \mid M(x)=1\right]=\mathbf{Pr}_{x\in D^+}\left[\hat{M}(x)=0\right]\leq \epsilon/4.$$

Also, $\hat{M}$ logically implies M. Since

$$\begin{aligned}\alpha &= \mathbf{Pr}_{(x,b)\in EX}[b=1 \mid M(x)=1]\\ &= \mathbf{Pr}_{(x,b)\in EX}\left[b=1 \mid \hat{M}(x)=1\right]\cdot \mathbf{Pr}_{x\in D}\left[\hat{M}(x)=1 \mid M(x)=1\right],\end{aligned}$$

it follows that $\alpha \geq \alpha' \geq \alpha(1-\epsilon/4)\geq \alpha-\epsilon/4$, and so $|\alpha-\hat{\alpha}|\leq 3\epsilon/4$. Thus, again making use of the fact that $\hat{M}$ has one-sided error, it can be seen that

$$\begin{aligned}\mathbf{E}_{x\in D}[|h(x)-c(x)|] &\leq \mathbf{Pr}_{x\in D}\left[\hat{M}(x)=0\wedge M(x)=1\right]+|\alpha-\hat{\alpha}|\cdot\mathbf{Pr}_{x\in D}\left[\hat{M}(x)=1\right]\\ &\leq \mathbf{Pr}_{x\in D}\left[\hat{M}(x)=0 \mid M(x)=1\right]+|\alpha-\hat{\alpha}|\\ &\leq \epsilon.\end{aligned}$$

■

Finally, we remark that Kearns, Schapire and Sellie [18] have recently extended this result beyond the class of partially visible monomials to the class of partially visible k-term DNF formulas. Specifically, if f is a k-term DNF formula over a set of hidden and visible variables, then Kearns, Schapire and Sellie give an efficient algorithm for learning with a model of probability the p-concept induced by regarding f as a probabilistic function over only the visible variables. (This assumes that the random assignment to the hidden variables is chosen independently of the assignment to the visible variables.) Their procedure uses as a subroutine the algorithm of Section 3.2 for learning probabilistic decision lists with increasing probabilities.

4 Hypothesis testing and expected loss

In this section, we address the problem of *hypothesis testing* in the p-concept model. More precisely, given a labeled sample, and a hypothesis p-concept, how do we decide how good h is with respect to the sample? As will be seen, the answer to this question depends on what our goal is (a decision rule or a model of probability).

We begin with a description of the learning framework that was proposed by Haussler [**?**], and that extends the work of Pollard [23], Dudley [10], Vapnik [28] and others. In this framework, the learner observes pairs (x,y) drawn randomly from some product space $X\times Y_0$ according to some fixed distribution. For instance, in the p-concept model, X is the domain, and $Y_0=\{0,1\}$; the target distribution on X and the target p-concept together induce a distribution on the space $X\times Y_0$.

Roughly speaking, in Haussler's model, the learner tries to find a hypothesis that accurately predicts the y-value of a random pair (x, y), given only the observed x-value. Thus, the hypothesis h should be such that $h(x)$ is "near" y for most random pairs (x, y).

It is often convenient not to restrict the range of h to the set Y_0; for instance, if $Y_0 = \{0, 1\}$, then we may want to allow h to map into $[0, 1]$. In general, then, we assume that h is a function which maps X into some set $Y \supset Y_0$.

In Haussler's model, the learner must choose a hypothesis from some given hypothesis space $\mathcal{H}$ of functions (each mapping X into Y). The goal of the learner is to find the hypothesis from $\mathcal{H}$ that minimizes the "discrepancy" on random pairs (x, y) between the observed value y, and the predicted value $h(x)$. This discrepancy between y and $h(x)$ is measured by a real-valued "loss" function. Formally, a *loss function* L is a function mapping $Y \times Y_0$ into $[0, 1]$. (The extension of such results to general bounded functions is straightforward.) Thus, the formal goal of the learner in this framework is to find a function $h \in \mathcal{H}$ that minimizes the average loss $\mathbf{E}[L(h(x), y)]$, where the expectation is over points (x, y) drawn randomly from $X \times Y_0$ according to the distribution on this product space.

Following Haussler [**?**], we adopt the notation $L_h(x, y) = L(h(x), y)$ for loss function L and hypothesis h. Moreover, we will write $\mathbf{E}[L_h]$ to denote the expected loss of h (with respect to L) under the unknown distribution on $X \times Y_0$. For a given sample $S = ((x_1, y_1), \ldots, (x_m, y_m))$ of m labeled examples, we will also be interested in the *empirical loss* of h:

$$\hat{\mathbf{E}}_S[L_h] = \frac{1}{m} \sum_{i=1}^{m} L_h(x_i, y_i).$$

Note that the empirical loss does not depend on the underlying distribution. Also, when the sample is clear from context, the subscript S is usually dropped.

We can cast the problems of learning decision rules and models of probability into this general framework. As mentioned above, in our setting $Y_0 = \{0, 1\}$ since an algorithm only sees $\{0, 1\}$-labels. For decision-rule learning, the algorithm outputs $\{0, 1\}$-valued hypotheses, and thus $Y = Y_0 = \{0, 1\}$ in this case. Similarly, for model-of-probability learning, we assume that hypotheses have range $[0, 1]$, and so $Y = [0, 1]$. The distribution on $X \times Y_0$ is naturally determined by the joint behavior of the target distribution D on X and the conditional probabilities $c(x)$ given by the target p-concept.

For finding the best decision rule, the *discrete* loss function is most appropriate, that is, the loss function Z given by the rule $Z(y, y') = 0$ if $y = y'$ and 1 otherwise. Then $\mathbf{E}[Z_h]$ is just the probability that h will misclassify a randomly drawn point, so minimizing $\mathbf{E}[Z_h]$ is equivalent to minimizing the predictive error.

For finding a model of probability, the *quadratic loss function* $Q(y, y') = (y - y')^2$ has some nice properties that make it the appropriate choice. These properties, which follow from the following theorem, are well known to statisticians. (See, for instance, White's review article [30].) Also, note that the empirical loss $\hat{\mathbf{E}}[Q_h]$ is the *average squared-error* statistic commonly used by researchers in pattern recognition and statistical decision theory.

Theorem 4.1 *For any target p-concept c, target distribution D, and p-concept h,*

$$\mathbf{E}[Q_h] - \mathbf{E}[Q_c] = \mathbf{E}_{x\in D}\left[(h(x) - c(x))^2\right].$$

Proof: For fixed $x \in X$, the probability that x is labeled 1 is $c(x)$, and in this case, h has loss

$$Q_h(x,1) = Q(h(x),1) = (1 - h(x))^2.$$

Likewise, x is labeled 0 with probability $1 - c(x)$, and in this case, h has loss $(h(x))^2$. Thus,

$$\mathbf{E}[Q_h] = \int_X \left[c(x)(1-h(x))^2 + (1-c(x))h(x)^2\right] dD(x).$$

Similarly,

$$\mathbf{E}[Q_c] = \int_X \left[c(x)(1-c(x))^2 + (1-c(x))c(x)^2\right] dD(x).$$

Applying straightforward algebra and linearity of integrals, it follows that

$$\begin{aligned}\mathbf{E}[Q_h] - \mathbf{E}[Q_c] &= \int_X [h(x) - c(x)]^2\, dD(x) \\ &= \mathbf{E}_{x\in D}\left[(h(x) - c(x))^2\right]\end{aligned}$$

as desired.

(All these integrals are defined, assuming as usual that c and h are measurable.) ■

Combined with Theorem 2.3, this theorem immediately suggests a computationally efficient method of choosing a good model of probability from a small (polynomial-size) class of candidate hypotheses. Suppose that a learning algorithm A has done some initial sampling and computation and has produced a class $\mathcal{H}$ of hypotheses, one of which is a good model of probability. Then A may simply use the empirical loss $\hat{\mathbf{E}}[Q_h]$ on a large enough labeled sample (a second sample) as an accurate estimate of the true loss $\mathbf{E}[Q_h]$ for each $h \in \mathcal{H}$, and then output the hypothesis with the smallest empirical loss. This hypothesis h must have near minimal true loss, and so, by the preceding theorem and our assumption that $\mathcal{H}$ contains a good model of probability, h must itself be a good model of probability.

For instance, we can use this method to prove that any efficient algorithm in the p-concept model (whose running time may be polynomial in $1/\delta$) can be converted into one whose running time is only polynomial in $\log(1/\delta)$. More precisely, suppose that A is an algorithm that, with probability at least $1/2$, succeeds in finding a "good" decision rule or model of probability h. Then we can convert A into an algorithm that successfully finds a good hypothesis with probability at least $1-\delta$ in time polynomial in $\log(1/\delta)$. The idea is to simply run A repeatedly, say $t = O(\log(1/\delta))$ times, producing hypotheses $h_1, \ldots, h_t$. With high probability, one of these hypotheses is "good," and we can find the best one by hypothesis testing each h_i and outputting the one with the lowest discrete or quadratic loss. (This technique is due to Haussler et al. [12] who prove the analogous result for the deterministic PAC model.)

The remainder of this section describes another example of an efficient learning algorithm that employs the approach outlined above.

4.1 Probabilistic concepts of k relevant variables

For a p-concept c on n Boolean variables, we say that variable x_i is *relevant* if $c(x) \neq c(y)$ for two vectors x and y which differ only in their ith bit. We say that c is a *p-concept of k relevant variables* if c has only k relevant variables. Such p-concepts are good models of situations in which there are a small number of variables whose settings determine the probabilistic behavior in a possibly very complicated manner, but most variables have no influence on this behavior.

Theorem 4.2 *Let $k \geq 1$ be fixed. Then the class of all p-concepts of k relevant variables is polynomially learnable with a model of probability.*

Proof: For any set $I \subset \{1, \ldots, n\}$, we say that two assignments x and y in $\{0,1\}^n$ are *equivalent with respect to I* if $x_i = y_i$ for all $i \in I$. Then this equivalence relation partitions $\{0,1\}^n$ into $2^{|I|}$ equivalence classes, called *I-blocks*.

Let c be the target p-concept, and let I_* be the set of indices of the k relevant variables of c.

Our algorithm begins by drawing a sample S_1 of size $m_1 = O((2^k/\epsilon^3) \cdot \log(2^k/\delta))$. For each of the $\binom{n}{k}$ sets I of k indices, and for each I-block B, our algorithm obtains from S_1 an estimate $\hat{p}_B$ of $p_B = \mathbf{Pr}_{(x,b)\in EX}[b = 1 \mid x \in B]$. A hypothesis h_I is then defined by the rule $h_I(x) = \hat{p}_B$ for $x \in B$.

By our choice of m_1, it follows from Chernoff bounds (Lemma 2.4) that, with probability at least $1 - \delta/2$, a sample S_1 is chosen for which $|\hat{p}_B - p_B| \leq \epsilon/4$ for every I_*-block B which satisfies $\mathbf{Pr}_{x\in D}[x \in B] > \epsilon/2^{k+2}$. This implies that, with high probability,

$$\mathbf{E}_{x\in D}[|h_{I_*}(x) - c(x)|] = \sum_B \mathbf{Pr}_{x\in D}[x \in B] \cdot |\hat{p}_B - c(x)| \leq \epsilon/2$$

where the sum is taken over all I_*-blocks B. This bound follows from the fact that $c(x) = p_B$ for $x \in B$, and by breaking the sum into two parts according to whether $\mathbf{Pr}_{x\in D}[x \in B]$ exceeds or does not exceed $\epsilon/2^{k+2}$.

Next, our algorithm tests each hypothesis h_I; that is, an estimate $\hat{\mathbf{E}}[Q_{h_I}]$ is found from a sufficiently large sample S_2 that, with high probability, is within $\epsilon/4$ of $\mathbf{E}[Q_{h_I}]$. Specifically, this will be the case with probability at least $1 - \delta/2$ for all hypotheses h_I if we choose a sample S_2 of size $O((1/\epsilon^2) \cdot \log(n^k/\delta))$. The algorithm outputs the hypothesis $h = h_I$ with the minimum empirical loss. Then, applying Theorem 4.1, we have:

$$\begin{aligned}
\mathbf{E}_{x\in D}\left[(h(x) - c(x))^2\right] &= \mathbf{E}[Q_h] - \mathbf{E}[Q_c] \\
&\leq \hat{\mathbf{E}}[Q_h] - \mathbf{E}[Q_c] + \epsilon/4 \\
&\leq \hat{\mathbf{E}}[Q_{h_{I_*}}] - \mathbf{E}[Q_c] + \epsilon/4 \\
&\leq \mathbf{E}[Q_{h_{I_*}}] - \mathbf{E}[Q_c] + \epsilon/2
\end{aligned}$$

$$= \mathbf{E}_{x\in D}\left[(h_{I_*}(x) - c(x))^2\right] + \epsilon/2 \leq \epsilon.$$

Applying Theorem 2.3, it follows that this efficient algorithm can be used to learn a good model of probability. ■

5 Uniform convergence methods

When is minimization of the empirical loss over a hypothesis class $\mathcal{H}$ sufficient to insure good learning of a decision rule or a model of probability? Note that even with computational issues set aside, the hypothesis-testing methods of the preceding section fall apart in the case of an infinite class $\mathcal{H}$: directly estimating the empirical loss of each $h \in \mathcal{H}$ separately would take an infinite number of examples and an infinite amount of time. What is required is a characterization of the number of examples required for uniform convergence of empirical losses to expected losses analogous to that provided by the VC-dimension in the case of deterministic concepts. This is particularly pressing in our model of p-concepts, where even when the domain is finite (e.g. $\{0,1\}^n$), the target p-concept class is usually infinite due to the different values allowed for the probabilities. We now turn to a discussion of such uniform convergence techniques applicable to p-concept classes.

Haussler [?], Pollard [23] and others have described the *pseudo dimension* of a class of real-valued functions $\mathcal{F}$ on domain X, and have shown that the pseudo dimension is a powerful tool for obtaining uniform convergence results.

Specifically, the pseudo dimension of $\mathcal{F}$ is defined as follows: Let $T = \{(x_1, r_1), \ldots, (x_d, r_d)\}$ be a set of d pairs, where each $x_i \in X$ and each $r_i \in \mathbb{R}$. We say that $\mathcal{F}$ *shatters* T if for every string $v \in \{0,1\}^d$ there is a function $f \in \mathcal{F}$ such that for $1 \leq i \leq d$, if $v_i = 0$ then $f(x_i) \leq r_i$ and if $v_i = 1$ then $f(x_i) > r_i$. Thus on the points $x_1, \ldots, x_d$ the class $\mathcal{F}$ exhibits all 2^d possible "above-below" behaviors with respect to the r_i. A geometric interpretation of this definition is to regard $(r_1, \ldots, r_d)$ as the origin of a coordinate system in d-dimensional Euclidean space; then $\mathcal{F}$ shatters T if the set $\{(f(x_1), \ldots, f(x_d)) : f \in \mathcal{F}\}$ intersects all 2^d orthants of the coordinate system. For this reason we will sometimes refer to $(r_1, \ldots, r_d)$ as the *origin of shattering.* The *pseudo dimension* of $\mathcal{F}$, denoted *PD*($\mathcal{F}$), is defined as the largest value of d for which there exists some set T of d pairs that is shattered by $\mathcal{F}$; if no such d exists, then *PD*($\mathcal{F}$) is infinite.

For us, the most important property of the pseudo dimension is that it allows us to upper bound the size of a sample sufficient to guarantee uniform convergence of empirical estimates for an entire class of functions. We state this formally in the following theorem which is adapted directly from Haussler's Corollary 2 [?].

For a hypothesis space $\mathcal{H}$ and loss function L, we define $L_{\mathcal{H}} = \{L_h : h \in \mathcal{H}\}$.

Theorem 5.1 *Let $\mathcal{H}$ be a hypothesis space of functions mapping X into Y which satisfies certain "permissibility" assumptions (see Haussler's paper). Let D be a probability distribution on $X \times Y_0$, let $L : Y \times Y_0 \to [0,1]$ be a loss function, let $d < \infty$ be the pseudo dimension of $L_{\mathcal{H}}$, and let S be a* *sample of m points from $X \times Y_0$ chosen randomly according to D. Assume*

$$m \geq m(d, \epsilon, \delta) = \frac{64}{\epsilon^2}\left(2d \ln\left(\frac{16e}{\epsilon}\right) + \ln\left(\frac{8}{\delta}\right)\right).$$

Then

$$\mathbf{Pr}[\exists h \in \mathcal{H} : \left|\hat{\mathbf{E}}[L_h] - \mathbf{E}[L_h]\right| > \epsilon] \leq \delta,$$

where the probability is taken over the random generation of S according to D.

Theorem 5.1 suggests the following canonical algorithm for finding a hypothesis from $\mathcal{H}$ with near minimum loss, when the pseudo dimension d is finite: take a sample S of at least $m(d, \epsilon/2, \delta)$ labeled examples from the oracle EX, and output any $h \in \mathcal{H}$ that minimizes the empirical loss $\hat{\mathbf{E}}[L_h]$ with respect to S. Then the theorem guarantees that the output hypothesis has true loss within ϵ of the best possible with probability at least $1-\delta$. This of course ignores the computational problem of actually finding such a hypothesis.

We can apply Theorem 5.1 to our learning problems by determining what the pseudo dimension is for each of the loss functions Z and Q. For the loss function Z, Haussler points out that the pseudo dimension is just the VC-dimension of the hypothesis class. That is, if $\mathcal{H}$ is a hypothesis space of functions with range $\{0,1\}$, then the pseudo dimension of the set of functions $Z_{\mathcal{H}}$ is just the VC-dimension of $\mathcal{H}$. Thus, the number of examples needed for decision-rule learning is bounded by the VC-dimension of the space of hypotheses used by the learning algorithm. (That the VC-dimension can be used in this manner was also observed by Blumer et al. [6].)

For example, consider the problem of learning a decision rule for an increasing function over $\mathbb{R}$. Note that the best decision rule for such a p-concept is always of the form $h_a(x) = 1$ for $x > a$, and 0 otherwise, for some a. Thus, a natural and efficient decision-rule learning algorithm for this problem is the following: draw a "large" sample from EX. Then, for each x_i in the sample, determine the empirical predictive error of hypothesis h_{x_i}, that is, the fraction of points in the sample whose labels disagree with h_{x_i}. Finally, output that h_{x_i} with the minimum empirical predictive error. Since the VC-dimension of this class of decision rules is 1, it follows from Theorem 5.1 that a polynomial-size sample suffices to insure the correctness of this algorithm.

For the problem of learning a model of probability, we will be interested in characterizing the pseudo dimension of $L_{\mathcal{H}}$ when L is the quadratic loss function Q, and $\mathcal{H}$ is a class of p-concepts over domain X. In fact, the following theorem shows that, in the p-concept model, the pseudo dimension of $Q_{\mathcal{H}}$ is equal to the pseudo dimension of $\mathcal{H}$.

Theorem 5.2 *For any p-concept class $\mathcal{H}$, the pseudo dimension of $Q_{\mathcal{H}}$ is equal to the pseudo dimension of $\mathcal{H}$.*

Proof: Let $\{(x_i, r_i)\}_{i=1}^{d}$ shatter $\mathcal{H}$. For all $v \in \{0,1\}^d$, there exists $h \in \mathcal{H}$ such that

$$sign(r_i - h(x_i)) = v_i,$$

where $sign(y) = 1$ if $y \geq 0$ and $sign(y) = 0$ if $y < 0$. Since all quantities are non-negative, $h(x_i) \leq r_i$ if and only if $Q_h(x_i, 0) = (h(x_i))^2 \leq r_i^2$. Thus,

$$v_i = sign(r_i^2 - Q_h(x_i, 0)),$$

and so $\{((x_i, 0), r_i)\}_{i=1}^d$ shatters $Q_{\mathcal{H}}$. Thus, the pseudo dimension of $Q_{\mathcal{H}}$ is at least $PD(\mathcal{H})$.

Conversely, let $\{((x_i, b_i), r_i)\}_{i=1}^d$ shatter $Q_{\mathcal{H}}$. Since d is finite, we can assume without loss of generality that the r_i's are chosen so that strict inequality holds in the definition of pseudo dimension, i.e., for all $v \in \{0,1\}^d$ there exists $h \in \mathcal{H}$ such that $Q_h(x_i, b_i) < r_i$ if $v_i = 1$ and $Q_h(x_i, b_i) > r_i$ if $v_i = 0$. Then

$$sign(r_i - Q_h(x_i, b_i)) = sign(r_i - (h(x_i) - b_i)^2) = sign(\sqrt{r_i} - |h(x_i) - b_i|)$$

which equals $sign(\sqrt{r_i} - h(x_i))$ if $b_i = 0$, and equals $sign(h(x_i) - (1 - \sqrt{r_i})) = 1 - sign((1 - \sqrt{r_i}) - h(x_i))$ if $b_i = 1$. It follows that $\{(x_i, |b_i - \sqrt{r_i}|)\}_{i=1}^d$ shatters $\mathcal{H}$, and thus the pseudo dimension of $Q_{\mathcal{H}}$ is at most $PD(\mathcal{H})$. ■

Note that the second part of the proof of this theorem relies critically on the fact that, in the p-concept model, instances are only $\{0,1\}$-labeled.

5.1 Linear function spaces

Armed with the definition of pseudo dimension and the sample size upper bounds provided by Theorem 5.1, we can now seek efficient algorithms that work by directly minimizing the quadratic loss over an infinite class of functions. This is the approach taken in our next theorem. For any domain X, let $f_i : X \to \mathbb{R}, 1 \leq i \leq d$ be any d functions, and let $\mathcal{C}(f_1, \ldots, f_d)$ denote the class of all p-concepts of the form $c(x) = \sum_{i=1}^d a_i f_i(x)$ for $a_i \in \mathbb{R}$, where we assume that the f_i and a_i are such that $c(x) \in [0,1]$ for all $x \in X$. We describe below an algorithm that learns a model of probability for p-concepts in the class $\mathcal{C}(f_1, \ldots, f_d)$. The running time of this algorithm is polynomial in the usual parameters, d, and the time needed to evaluate the functions f_i.

This result can be applied to prove the polynomial learnability of several natural p-concept classes. For instance, consider the generalization of deterministic disjunctions in which the target p-concept has the form $c(x) = (x_{i_1} + \cdots + x_{i_t})/t$, where the x_{i_j} are Boolean variables chosen from $x_1, \ldots, x_n$, and $+$ denotes ordinary addition. Thus, such a p-concept is "more positive" on vectors $x \in \{0,1\}^n$ that have many of the relevant variables set to 1. Such a p-concept class is clearly of the form required by Theorem 5.3, and so is polynomially learnable with a model of probability.

As a more subtle application, consider a class of p-concepts over $\{0,1\}^n$ that are partially specified by a canonical positive example $z \in \{0,1\}^n$. We wish to model a setting in which z is the prototypical positive instance, and those examples "most like" z are more likely to be labeled positively. Thus, the target p-concept might have the form $c(x) = a - b \cdot d(x, z)$ where $d(x, z)$ denotes the Hamming distance and a and b are positive real-valued coefficients such that c is maximized at z and is always in the range $[0,1]$. Here the p-concept class $\mathcal{C}$ is obtained by ranging over the choices of the prototype z and the coefficients a and b, and the "decay function," which specifies the rate at which vectors further away from the prototype fail to exemplify the concept, is linear. It is not difficult to show that each function in $\mathcal{C}$ can in fact be written as a weighted linear sum of the variables $x_1, \ldots, x_n$, so $\mathcal{C}$ is polynomially learnable with a model of probability.

Finally, we remark that Theorem 5.3 can be applied to learn so-called "t-transform functions" considered by Mansour [21].

Theorem 5.3 *For any set of d known computable functions $f_i : X \to \mathbb{R}, 1 \leq i \leq d$, the class $\mathcal{C}(f_1, \ldots, f_d)$ is learnable with a model of probability. Specifically, there exists a learning algorithm for this class whose running time is polynomial in $1/\epsilon$, $\log(1/\delta)$, $1/\gamma$, d and the maximum time needed to evaluate any of the functions f_i.*

Proof: Given $\epsilon, \delta > 0$, our algorithm draws a sample of size $m = \lceil m(d, \epsilon/2, \delta) \rceil$ as given by Theorem 5.1, and attempts to find the choice of coefficients $a_1, \ldots, a_d$ that minimizes the quadratic loss over the sample. This can be done using a standard least-squares approximation. For instance, this can be done directly by differentiating with respect to each unknown coefficient a_i the expression $\sum_{j=1}^{m} \left[\left(\sum_{i=1}^{d} a_i f_i(x_j) \right) - b_j \right]^2$ (where $\{(x_j, b_j)\}_{j=1}^{m}$ is the labeled sample), and setting the resulting partial derivative to 0. This yields a system of d linear equations in the d variables a_i that is of a special form and that can be solved using standard techniques. Cormen, Leiserson and Rivest [7, Chapter 31] describe in detail how this can be done efficiently; see also Duda and Hart [9].

Let $\hat{a}_1, \ldots, \hat{a}_d$ be the resulting solution, and let $h_0 = \sum_{i=1}^{d} \hat{a}_i f_i$. Note that h_0 may not be bounded between 0 and 1, and so may not be in $\mathcal{C} = \mathcal{C}(f_1, \ldots, f_d)$. We show below how to handle this difficulty.

For any real-valued function f, let $clamp(f)$ denote the function obtained by "clamping" f between 0 and 1; that is, $clamp(f) = g \circ f$ where $g : \mathbb{R} \to \mathbb{R}$, and $g(x)$ is defined to be 0 if $x \leq 0$, x if $0 \leq x \leq 1$, and 1 if $x \geq 1$. Let $\mathcal{H} = \left\{ clamp\left(\sum_{i=1}^{d} a_i f_i \right) : a_i \in \mathbb{R} \right\}$. Our algorithm outputs the hypothesis $h = clamp(h_0)$. Clearly h is in $\mathcal{H}$, as is the target c.

Dudley [10] shows that a d-dimensional linear function space has pseudo dimension d. (This is reproved by Haussler [**?**], Theorem 4.) Combined with Haussler's Theorem 5 (which concerns the pseudo dimension of families of functions constructed in the same way as $\mathcal{H}$), this immediately implies $PD(\mathcal{H}) \leq d$. Thus, by Theorem 5.1 and our choice of m, with probability at least $1 - \delta$, $\left| \mathbf{E}[Q_{h'}] - \hat{\mathbf{E}}[Q_{h'}] \right| \leq \epsilon/2$ for every $h' \in \mathcal{H}$. Also, note that $\hat{\mathbf{E}}[Q_h] \leq \hat{\mathbf{E}}[Q_{h_0}]$ since all instances in the sample are $\{0,1\}$-labeled, so clamping the hypothesis only improves its performance. Thus, with probability at least $1 - \delta$, we have:

$$\begin{aligned} \mathbf{E}_{x \in D}\left[(h(x) - c(x))^2 \right] &= \mathbf{E}[Q_h] - \mathbf{E}[Q_c] \\ &\leq \hat{\mathbf{E}}[Q_h] - \mathbf{E}[Q_c] + \epsilon/2 \\ &\leq \hat{\mathbf{E}}[Q_{h_0}] - \mathbf{E}[Q_c] + \epsilon/2 \\ &\leq \hat{\mathbf{E}}[Q_c] - \mathbf{E}[Q_c] + \epsilon/2 \leq \epsilon. \end{aligned}$$

As usual, Theorem 2.3 can be applied to convert this algorithm into one that learns a good model of probability for this class. ■

6 A lower bound on sample size

Theorem 5.1 provides a kind of general upper bound on the sample size required for learning a model of probability. We turn now to the problem of lower bounds on sample complexity in this framework. For this, we need to introduce a refined notion of shattering.

Let $\mathcal{H}$ be a class of p-concepts over domain X. Let $T = \{(x_1, r_1), \ldots, (x_d, r_d)\}$ be a set of d pairs, where each $x_i \in X$ and each $r_i \in [0,1]$. For $w > 0$, we say that $\mathcal{H}$ *w-shatters* T if for every string $v \in \{0,1\}^d$ there is a p-concept $h \in \mathcal{H}$ (a *witness*) such that for $1 \leq i \leq d$, if $v_i = 0$ then $h(x_i) < r_i - w$ and if $v_i = 1$ then $h(x_i) > r_i + w$. Thus, in addition to T being shattered by $\mathcal{H}$ we require that there be a *separation* of width w between r_i and $h(x_i)$ for each witness h; we call w the *width of shattering*. Note that if $\mathcal{H}$ has pseudo dimension at least d then there always exists some $w > 0$ such that some set of d pairs over $X \times [0,1]$ is w-shattered.

Based on this stronger notion of shattering, we can now prove the following lower bound on sample complexity in our model. This lower bound, combined with Theorems 5.1 and 5.2, shows that when the pseudo dimension is finite it characterizes the sample size required for learning with a model of probability (that is, the bound obtained by applying Theorem 5.1 is tight within a polynomial factor of $1/\epsilon$ and $1/\delta$). This lower bound may also be of theoretical interest, since in Haussler's general learning framework [**?**] the pseudo dimension is used only to approximately upper bound the so-called *covering number*, which is directly used to obtain sample-size bounds.

Theorem 6.1 *Let $\mathcal{C}$ be a p-concept class that w-shatters a set of cardinality d. Then for $\gamma \leq w$ and $\epsilon + \delta \leq 1/8$, any algorithm for learning $\mathcal{C}$ with a model of probability requires at least $\lfloor d(\lg e)/8 \rfloor = \Omega(d)$ examples.*

Proof: Our proof is based on the analogous lower bound proof given by Blumer et al. [6] for learning deterministic concepts. However, the analysis is more involved in the probabilistic case.

Let $T = \{(x_1, r_1), \ldots, (x_d, r_d)\}$ be w-shattered by the p-concept class $\mathcal{C}$. Let $\mathcal{C}_0 \subseteq \mathcal{C}$ be any fixed subclass of $\mathcal{C}$ such that $\mathcal{C}_0$ w-shatters T and $|\mathcal{C}_0| = 2^d$. Let A be a learning algorithm for $\mathcal{C}$ taking m examples for the given choices of ϵ, δ and γ, and let h_S denote the hypothesis output by A on input a labeled sample S of size m. (We assume for simplicity that A is deterministic — the proof is easily modified to handle randomized algorithms.)

Let the target distribution D be uniform over the points $x_1, \ldots, x_d$. We define a weak error measure $e(c, S)$ for target $c \in \mathcal{C}_0$ and input sample S as follows: the error $e_i(c, S)$ at x_i is defined to be 0 if $c(x_i)$ and $h_S(x_i)$ are either both less than r_i, or both greater than r_i; otherwise, $e_i(c, S) = 1$. Then e is just the average of the e_i's:

$$e(c, S) = \frac{1}{d} \sum_{i=1}^{d} e_i(c, S).$$

Note that if $e(c,S) > \epsilon$, then h_S cannot be an (ϵ, w)-good model of probability for c since if $c(x_i)$ and $h_S(x_i)$ are not "on the same side" of r_i, then they differ by more than w.

This definition allows us to examine the expectation

$$\mathbf{E}_S[e(c,S)] = \sum_S \mathbf{Pr}[S|c] \cdot e(c,S)$$

which is taken over S drawn randomly according to D and labeled randomly according to c, and $\mathbf{Pr}[S|c]$ is the conditional probability that S is generated by D and c.

We will also be interested in the expectation of $e(c,S)$ when both $c \in \mathcal{C}_0$ is generated uniformly at random and S is generated according to the randomly chosen c and the target distribution D:

$$\mathbf{E}_{c,S}[e(c,S)] = \frac{1}{2^d} \sum_S \sum_{c \in \mathcal{C}_0} \mathbf{Pr}[S|c] \cdot e(c,S).$$

We wish to lower bound $e(c,S)$ for most of the p-concepts in $\mathcal{C}_0$. For any sample S, let $\mathcal{C}_S = \{c \in \mathcal{C}_0 : e(c,S) < 1/4\}$. Then for $c \in \mathcal{C}_0 - \mathcal{C}_S$, $e(c,S) \geq 1/4$, and so we obtain the lower bound

$$\mathbf{E}_{c,S}[e(c,S)] \geq \frac{1}{2^{d+2}} \sum_S \sum_{c \in \mathcal{C}_0 - \mathcal{C}_S} \mathbf{Pr}[S|c] = \frac{1}{2^{d+2}} \left(\sum_S \sum_{c \in \mathcal{C}_0} \mathbf{Pr}[S|c] - \sum_S \sum_{c \in \mathcal{C}_S} \mathbf{Pr}[S|c] \right).$$

Now

$$\sum_S \sum_{c \in \mathcal{C}_0} \mathbf{Pr}[S|c] = \sum_{c \in \mathcal{C}_0} \sum_S \mathbf{Pr}[S|c] = |\mathcal{C}_0| = 2^d$$

since for any c, $\sum_S \mathbf{Pr}[S|c] = 1$. To upper bound $\sum_{c \in \mathcal{C}_0} \sum_S \mathbf{Pr}[S|c]$, we will first derive upper bounds on the total number of possible samples S, the cardinality of $\mathcal{C}_S$, and the value of $\mathbf{Pr}[S|c]$. First, the number of possible samples S is at most $(2d)^m$, since each of the d points may appear with either label and the number of examples in S is m. To bound $|\mathcal{C}_S|$, consider drawing a p-concept c uniformly at random from the class $\mathcal{C}_0$. By choice of $\mathcal{C}_0$, the probability that $e(c,S) < 1/4$ is bounded by the probability of fewer than $d/4$ heads occurring in d flips of a fair coin. Thus, applying Chernoff bounds (Lemma 2.4), we conclude

$$|\mathcal{C}_S| \leq |\mathcal{C}_0| \cdot e^{-d/8} = 2^{(1-a_0)d}$$

where $a_0 = (\lg e)/8$. Finally, $\mathbf{Pr}[S|c] \leq 1/d^m$ since if we ignore the labels on the points in S, the probability of any particular set of m points being generated by the target distribution D is at most $1/d^m$.

Piecing together these bounds, we may now write

$$\mathbf{E}_{c,S}[e(c,S)] \geq \frac{1}{2^{d+2}} \left(2^d - (2d)^m \cdot 2^{(1-a_0)d} \cdot d^{-m} \right) = \frac{1}{4} \left(1 - 2^{m - a_0 d} \right).$$

Thus, if $m \leq a_0 d - 1$ then $\mathbf{E}_{c,S}[e(c,S)] \geq 1/8$. From this it follows that for some fixed $c_0 \in \mathcal{C}_0$, $\mathbf{E}_S[e(c_0,S)] \geq 1/8$ where the expectation is taken over S drawn according to D and labeled according to c_0. By assumption A learns with a model of probability. Thus, with probability at least $1-\delta$, a sample S is chosen such that h_S is an (ϵ, w)-good model of probability. As noted above, in such a case, $e(c,S) \leq \epsilon$. Thus, $\mathbf{E}_S[e(c_0,S)] \leq (1-\delta)\epsilon + \delta < \epsilon + \delta$. Therefore, if $\epsilon+\delta \leq 1/8$ then m is at least $\lfloor a_0 d \rfloor$, proving the theorem. ■

Any theorem giving a sample-size lower bound must incorporate the width of shattering; for instance, ours holds only for $\gamma \leq w$. To see that this is necessary, note that the p-concept class of all functions mapping X into $\{1/2 - w, 1/2 + w\}$ shatters all of X, but for $\gamma \geq w$ this class can be learned with *no* examples with the hypothesis $h(x) \equiv 1/2$. A more natural example is provided by the non-decreasing functions of Section 3.1. Here the pseudo dimension is infinite, but we have an efficient learning algorithm. An interesting open problem is to give improved general upper bounds on sample size that incorporate the width of shattering.

7 Occam's Razor for general loss functions

In this section, we present a generalized form of Occam's Razor [5] applicable to the minimization of bounded loss functions, and in particular to learning p-concepts with a model of probability or a decision rule. Here we have several motivations: first, it is of philosophical interest to investigate the most general conditions under which learning is equivalent to some form of data compression; second, as in the Valiant model, we hope that Occam's Razor will help isolate and simplify the probabilistic analysis of learning algorithms; third, Occam's Razor may be easier to apply than uniform-convergence methods in the case that the pseudo dimension is unknown or difficult to compute; and fourth, Occam's Razor may give better sample-size bounds than direct analyses.

An *Occam algorithm* for hypothesis class $\mathcal{H}$ over parametrized domain X, with respect to a loss function $L : Y \times Y_0 \rightarrow [0,1]$ is a polynomial-time algorithm A that takes as input a labeled sample $S \in (X_n \times Y_0)^m$, and outputs a hypothesis h with the properties that:

1. $\hat{\mathbf{E}}[L_h] - \inf_{h' \in \mathcal{H}} \hat{\mathbf{E}}[L_{h'}] \leq \tau = \tau(n,m) = n^a m^{-\alpha}$ for some constants $a \geq 0$, and $\alpha > 0$; and
2. h can be represented by a string over the finite alphabet $\{0,1\}$ of encoded length $\ell = \ell(n,m) = n^b m^\beta$ for some constants $b \geq 0$ and $\beta < 1$.

Thus, as in the non-probabilistic setting, we require an Occam algorithm to perform some kind of data compression, i.e., to output a hypothesis significantly smaller than the given sample. Furthermore, the output hypothesis must come close to minimizing the empirical loss on the sample over the entire hypothesis space $\mathcal{H}$.

Theorem 7.1 *Let A be an Occam algorithm as described above. Let S be a labeled sample of size m generated according to some target p-concept c. Let h be the result of running A on S. Assume m is so large that $\tau \leq \epsilon/4$ and $2(2^\ell + 1)e^{-\epsilon^2 m/8} \leq \delta$. Then*

$$\mathbf{Pr}[\mathbf{E}[L_h] - \inf_{h' \in \mathcal{H}} \mathbf{E}[L_{h'}] > \epsilon] \leq \delta.$$

In particular, this will be the case if

$$m \geq \max\left\{\left(\frac{4n^a}{\epsilon}\right)^{1/\alpha}, \left(\frac{16(\ln 2)n^b}{\epsilon^2}\right)^{1/(1-\beta)}, \frac{16\ln(4/\delta)}{\epsilon^2}\right\}.$$

Proof: The proof is analogous to that of Blumer et al. [5].

Let $\mathcal{H}_A$ be the set of (at most) 2^ℓ hypotheses which might potentially be output by A. Let $h_* \in \mathcal{H}$ be such that $\mathbf{E}[L_{h_*}] \leq \inf_{h' \in \mathcal{H}} E[L_{h'}] + \epsilon/4$. Then, by Chernoff bounds (Lemma 2.4), the probability that either $\mathbf{E}[L_{h'}] > \hat{\mathbf{E}}[L_{h'}] + \epsilon/4$ for any $h' \in \mathcal{H}_A$, or that $\hat{\mathbf{E}}[L_{h_*}] > E[L_{h_*}] + \epsilon/4$ is at most $2(2^\ell + 1)e^{-\epsilon^2 m/8} \leq \delta$. So, with probability at least $1 - \delta$,

$$\begin{aligned} \mathbf{E}[L_h] &\leq \hat{\mathbf{E}}[L_h] + \epsilon/4 \\ &\leq \inf_{h' \in \mathcal{H}} \hat{\mathbf{E}}[L_{h'}] + \epsilon/2 \\ &\leq \hat{\mathbf{E}}[L_{h_*}] + \epsilon/2 \\ &\leq \mathbf{E}[L_{h_*}] + 3\epsilon/4 \\ &\leq \inf_{h' \in \mathcal{H}} \mathbf{E}[L_{h'}] + \epsilon. \end{aligned}$$

We show next that the stated bound on m is sufficient. Clearly, from the first bound on m, $\tau \leq \epsilon/4$. Further, from the second bound, we have that $\ell = n^b m^\beta \leq (\lg e)\epsilon^2 m/16$. Thus, $2(2^\ell + 1)e^{-\epsilon^2 m/8} \leq 4 \cdot 2^\ell e^{-\epsilon^2 m/8} \leq 4 \cdot e^{-\epsilon^2 m/16} \leq \delta$ by the last bound on m. ■

As an example, Theorem 7.1 can be applied to the problem of learning p-concepts with k relevant variables. Essentially, the algorithm given in Theorem 4.2 can be modified so that a single initial sample of size m can be used for all of the estimates made by that algorithm. Note that a hypothesis output by this algorithm can be represented by the names of k of the variables, plus the probabilities for the 2^k equivalence classes. Each name requires $\lg n$ bits, and moreover, each probability is a rational number (being an empirical probability estimate) that requires only $O(\log m)$ bits; thus, the hypothesis has size $O(k \log n + 2^k \log m)$. Finally, it can be shown that the hypothesis has the minimum empirical loss over the *entire* class of p-concepts with k relevant variables. Thus, Theorem 7.1 can be used to easily determine an appropriate sample size for this algorithm.

Note that Theorem 7.1 is only applicable to algorithms which output hypotheses over a finite alphabet. However, the theorem can be extended to apply to other algorithms in a manner similar to the approach taken by Littlestone and Warmuth [20] in the Valiant model. The basic idea is to allow the learning algorithm to output hypotheses that can be represented over the alphabet $S \cup \{0, 1\}$, where S is the given sample. That is, the representation of the hypothesis may include individual examples from the sample itself. For example, the hypothesis output by the algorithm for learning increasing functions with a decision rule (Section 5) can be represented by a single example from the sample, despite the fact that this hypothesis would require an infinite number of bits to represent over a fixed finite alphabet. Thus, this alternate form of Occam's Razor can be used to provide a good sample-size bound. Similarly, the algorithm given in Theorem 3.1 (slightly modified) for learning increasing functions with a model of probability can be cast in this light as an Occam algorithm.

8 Conclusions and open problems

In this paper, we have explored an extension of Valiant's model that incorporates the uncertainty inherent in many real-world learning problems. We have focused primarily on techniques for the design of efficient algorithms in this model.

Naturally, we would like to find efficient algorithms for much broader classes of p-concepts than the simple classes considered here. For example, can the algorithm of Section 3.2 be extended to learn arbitrary (not necessarily ω-converging) probabilistic decision lists? As is often the case in the deterministic Valiant model, sample size is not the problem: from Theorem 7.1, one can fairly easily derive a polynomial sample-size bound for learning this class using a computationally inefficient Occam algorithm that, given a sample, finds the decision list with the minimum quadratic loss by trying all permutations of the list order. The problem here is computational: how can we learn this class efficiently? The development of further techniques for learning p-concepts is a vitally important direction for further research.

Although the p-concept model captures realistic aspects of many learning problems, it might still be criticized for its assumption that the target p-concept belongs to an a priori known class of p-concepts. More realistic is a so-called *agnostic* learning model in which the target p-concept is *any* function from X into $[0, 1]$, and the learner's goal is to find the best hypothesis from some fixed space of hypotheses. This is actually the framework assumed by Haussler [**?**] in deriving his sample-size bounds. A few of the algorithms described in this paper are effective agnostic learners, such as the algorithm of Theorem 4.2 for p-concepts with k relevant variables. An important open problem is the extension of other algorithms to agnostic learning. For instance, do there exist efficient agnostic algorithms for probabilistic decision lists with ω-converging probabilities (Section 3.2), or for linear function spaces (Section 5.1)?

It is also important to continue to develop a theoretical foundation for p-concept learning. For instance, are there other loss functions that might be appropriate, such as the log loss function? (See Haussler [**?**] in this regard.) Also, can the lower bound proof of Theorem 6.1 be significantly improved?

Finally, consistent with our quest for efficient algorithms is the need to be able to recognize that a learning problem is computationally intractable. Various techniques in this regard have been developed in the Valiant model, such as those of Pitt and Valiant [22], and Kearns and Valiant [17, 15]. Can such techniques be extended to the p-concept model? Both of these results seem to depend crucially on the deterministic nature of the Valiant model. What then would a negative, computational result look like in the p-concept model?

Acknowledgements

We are deeply indebted to David Haussler and Ron Rivest for their considerable help and guidance in preparing this paper. We also wish to thank Avrim Blum, Yoav Freund, Umesh Vazirani and two anonymous referees for their comments and suggestions.

Financial support for this paper was provided through ARO Grant DAAL03-86-K-0171, DARPA Contract N00014-89-J-1988, NSF Grant CCR-8914428, and a grant from the Siemens Corporation.

References

[1] Naoki Abe, Jun-ichi Takeuchi, and Manfred K. Warmuth. Polynomial learnability of probabilistic concepts with respect to the Kullback-Liebler divergence. In *Proceedings of the Fourth Annual Workshop on Computational Learning Theory*, pages 277–289, August 1991.

[2] William Aiello and Milena Mihail. Learning the Fourier spectrum of probabilistic lists and trees. In *Proceedings of the Second Annual ACM-SIAM Symposium on Discrete Algorithms*, January 1991.

[3] Dana Angluin and Philip Laird. Learning from noisy examples. *Machine Learning*, 2(4):343–370, 1988.

[4] Dana Angluin and Leslie G. Valiant. Fast probabilistic algorithms for Hamiltonian circuits and matchings. *Journal of Computer and System Sciences*, 18(2):155–193, April 1979.

[5] Anselm Blumer, Andrzej Ehrenfeucht, David Haussler, and Manfred K. Warmuth. Occam's razor. *Information Processing Letters*, 24(6):377–380, April 1987.

[6] Anselm Blumer, Andrzej Ehrenfeucht, David Haussler, and Manfred K. Warmuth. Learnability and the Vapnik-Chervonenkis dimension. *Journal of the Association for Computing Machinery*, 36(4):929–965, October 1989.

[7] Thomas H. Cormen, Charles E. Leiserson, and Ronald L. Rivest. *Introduction to Algorithms.* MIT Press, 1990.

[8] Annette J. Dobson. *An Introduction to Generalized Linear Models.* Chapman and Hall, 1990.

[9] Richard O. Duda and Peter E. Hart. *Pattern Classification and Scene Analysis.* Wiley, 1973.

[10] R. M. Dudley. Central limit theorems for empirical measures. *The Annals of Probability*, 6(6):899–929, 1978.

[11] David Haussler. Generalizing the PAC model: Sample size bounds from metric dimension-based uniform convergence results. In *30th Annual Symposium on Foundations of Computer Science*, pages 40–45, October 1989.

[12] David Haussler, Michael Kearns, Nick Littlestone, and Manfred K. Warmuth. Equivalence of models for polynomial learnability. *Information and Computation*, 95(2):129–161, December 1991.

[13] Wassily Hoeffding. Probability inequalities for sums of bounded random variables. *Journal of the American Statistical Association*, 58(301):13–30, March 1963.

[14] Abraham Kandel. *Fuzzy Techniques in Pattern Recognition.* Wiley, 1982.

[15] Michael Kearns. *The Computational Complexity of Machine Learning.* MIT Press, 1990.

[16] Michael Kearns and Ming Li. Learning in the presence of malicious errors. In *Proceedings of the Twentieth Annual ACM Symposium on Theory of Computing*, pages 267–280, May 1988. To appear, *SIAM Journal on Computing.*

[17] Michael Kearns and Leslie G. Valiant. Cryptographic limitations on learning Boolean formulae and finite automata. In *Proceedings of the Twenty First Annual ACM Symposium on Theory of Computing*, pages 433–444, May 1989. To appear, *Journal of the Association for Computing Machinery.*

[18] Michael J. Kearns, Robert E. Schapire, and Linda M. Sellie. Toward efficient agnostic learning. In *Proceedings of the Fifth Annual ACM Workshop on Computational Learning Theory*, pages 341–352, July 1992.

[19] Nathan Linial, Yishay Mansour, and Ronald L. Rivest. Results on learnability and the Vapnik-Chervonenkis dimension. In *29th Annual Symposium on Foundations of Computer Science*, pages 120–129, October 1988.

[20] Nick Littlestone and Manfred Warmuth. Relating data compression and learnability. Unpublished manuscript, November 1987.

[21] Yishay Mansour. Learning via Fourier transform. Unpublished manuscript, April 1990.

[22] Leonard Pitt and Leslie G. Valiant. Computational limitations on learning from examples. *Journal of the Association for Computing Machinery*, 35(4):965–984, October 1988.

[23] David Pollard. *Convergence of Stochastic Processes.* Springer-Verlag, 1984.

[24] Ronald L. Rivest. Learning decision lists. *Machine Learning*, 2(3):229–246, 1987.

[25] Yasubumi Sakakibara. *Algorithmic Learning of Formal Languages and Decision Trees.* PhD thesis, Tokyo Institute of Technology, October 1991. Research Report IIAS-RR-91-22E, International Institute for Advanced Study of Social Information Science, Fujitsu Laboratories, Ltd.

[26] Robert H. Sloan. Types of noise in data for concept learning. In *Proceedings of the 1988 Workshop on Computational Learning Theory*, pages 91–96, August 1988.

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

[28] V. N. Vapnik. *Estimation of Dependences Based on Empirical Data.* Springer-Verlag, 1982.

[29] V. N. Vapnik and A. Ya. Chervonenkis. On the uniform convergence of relative frequencies of events to their probabilities. *Theory of Probability and its applications*, XVI(2):264–280, 1971.

[30] Halbert White. Learning in artificial neural networks: A statistical perspective. *Neural Computation*, 1(4):425–464, 1989.

[31] Kenji Yamanishi. A learning criterion for stochastic rules. In *Proceedings of the Third Annual Workshop on Computational Learning Theory*, pages 67–81, August 1990. To appear, *Machine Learning.*

[32] L. A. Zadeh. Fuzzy sets. *Information and Control*, 8(3):338–353, June 1965.