Polynomial regression under arbitrary product distributions

Eric Blais and **Ryan O'Donnell** and **Karl Wimmer**
Carnegie Mellon University

Abstract

In recent work, Kalai, Klivans, Mansour, and Servedio [KKMS05] studied a variant of the "Low-Degree (Fourier) Algorithm" for learning under the uniform probability distribution on $\{0,1\}^n$. They showed that the L_1 polynomial regression algorithm yields *agnostic* (tolerant to arbitrary noise) learning algorithms with respect to the class of threshold functions — under certain restricted instance distributions, including uniform on $\{0,1\}^n$ and Gaussian on $\mathbb{R}^n$. In this work we show how *all* learning results based on the Low-Degree Algorithm can be generalized to give almost identical agnostic guarantees under *arbitrary* product distributions on instance spaces $X_1 \times \cdots \times X_n$. We also extend these results to learning under *mixtures* of product distributions.

The main technical innovation is the use of (Hoeffding) orthogonal decomposition and the extension of the "noise sensitivity method" to arbitrary product spaces. In particular, we give a very simple proof that threshold functions over arbitrary product spaces have δ-noise sensitivity $O(\sqrt{\delta})$, resolving an open problem suggested by Peres [Per04].

1 Introduction

In this paper we study binary classification learning problems over arbitrary instance spaces $\mathcal{X} = X_1 \times \cdots \times X_n$. In other words, each instance has n "categorical attributes", the ith attribute taking values in the set X_i. For now we assume that each X_i has cardinality at most $\mathrm{poly}(n)$.[1]

It is convenient for learning algorithms to encode instances from $\mathcal{X}$ as vectors in $\{0,1\}^{|X_1|+\cdots+|X_n|}$ via the "one-out-of-k encoding"; e.g., an attribute from $X_1 = \{\text{red, green, blue}\}$ is replaced by one of $(1,0,0)$, $(0,1,0)$, or $(0,0,1)$. Consider now the following familiar learning algorithm:

> Given m examples of training data $(\vec{x}_1, y_1), \ldots, (\vec{x}_m, y_m) \in \mathcal{X} \times \{-1,1\}$,
>
> 1. Expand each instance $\vec{x}_i$ into a vector from $\{0,1\}^{|X_1|+\cdots+|X_n|}$ via the "one-out-of-k" encoding.
> 2. Consider "features" which are products of up to d of the new 0-1 attributes.
> 3. Find the linear function W in the feature space that best fits the training labels under some loss measure ℓ: e.g., squared loss, hinge loss, or L_1 loss.
> 4. Output the hypothesis $\mathrm{sgn}(W - \theta)$, where $\theta \in [-1,1]$ is chosen to minimize the hypothesis' training error.

We will refer to this algorithm as "degree-d polynomial regression (with loss ℓ)". When ℓ is the hinge loss, this is equivalent to the soft margin SVM algorithm with the degree-d polynomial kernel and no regularization [CV95].[2] When ℓ is the squared loss and the data is drawn i.i.d. from the uniform distribution on $\mathcal{X} = \{0,1\}^n$, the algorithm is effectively equivalent to the Low-Degree Algorithm of Linial, Mansour, and Nisan [LMN93] — see [KKMS05]. Using techniques from convex optimization (indeed, linear programming for L_1 or hinge loss, and just basic linear algebra for squared loss), it is known that the algorithm can be performed in time $\mathrm{poly}(m, n^d)$. For all known proofs of good generalization for the algorithm, $m = n^{\Theta(d)}/\epsilon$ training examples are necessary (and sufficient). Hence we will view the degree-d polynomial regression algorithm as requiring $\mathrm{poly}(n^d/\epsilon)$ time and examples. (Because of this, whether or not one uses the "kernel trick" is a moot point.)

Although SVM-based algorithms are very popular in practice, the scenarios in which they *provably* learn successfully are relatively few (see Section 1.2 below) — especially when there is error in the labels. Our goal in this paper is to broaden the class of scenarios in which learning with polynomial regression has provable, polynomial-time guarantees.

[1] Given real-valued attributes, the reader may think of bucketing them into $\mathrm{poly}(n)$ buckets.

[2] Except for the minor difference of choosing an optimal θ rather than fixing $\theta = 0$.

1.1 The learning framework

We study binary classification learning in the natural "agnostic model" [KSS94] (sometimes described as the model with arbitrary classification noise). We assume access to training data drawn i.i.d. from some distribution $\mathcal{D}$ on $\mathcal{X}$, where the labels are provided by an arbitrary unknown "target" function $t : \mathcal{X} \to \{-1, 1\}$. The task is to output a hypothesis $h : \mathcal{X} \to \{-1, 1\}$ which is a good predictor on future examples from $\mathcal{D}$. We define the "error of h" to be $\mathrm{err}(h) = \mathbf{Pr}_{\boldsymbol{x}\sim\mathcal{D}}[h(\boldsymbol{x}) \neq t(\boldsymbol{x})]$.[3] We compare the error of an algorithm's hypothesis with the best error achievable among functions in a fixed class $\mathcal{C}$ of functions $\mathcal{X} \to \{-1, 1\}$. Define $\mathrm{Opt} = \inf_{f\in\mathcal{C}} \mathrm{err}(f)$. We say that an algorithm $\mathcal{A}$ "agnostically learns with respect to $\mathcal{C}$" if, given $\epsilon > 0$ and access to training data, it outputs a hypothesis h which satisfies $\mathbf{E}[\mathrm{err}(h)] \leq \mathrm{Opt} + \epsilon$. Here the expectation is with respect to the training data drawn.[4] The running time (and number of training examples) used are measured as functions of n and ϵ.

Instead of an instance distribution $\mathcal{D}$ on $\mathcal{X}$ and a target $t : \mathcal{X} \to \{-1, 1\}$, one can more generally allow a distribution $\mathcal{D}'$ on $\mathcal{X}\times\{-1, 1\}$; in this case, $\mathrm{err}(h) = \mathbf{Pr}_{(\boldsymbol{x},\boldsymbol{y})\sim\mathcal{D}'}[h(\boldsymbol{x}) \neq \boldsymbol{y}]$. Our learning results also hold in this model just as in [KKMS05]; however we use the simpler definition for ease of presentation, except in Section 5.3.

In the special case when t is promised to be in $\mathcal{C}$ we are in the scenario of PAC learning [Val84]. This corresponds to the case $\mathrm{Opt} = 0$. Since $\mathcal{C}$ is usually chosen (by necessity) to be a relatively simple class, the PAC model's assumption that there is a perfect classifier in $\mathcal{C}$ is generally considered somewhat unrealistic. This is why we work in the agnostic model.

Finally, since strong hardness results are known [KSS94, LBW95, KKMS05, GR06] for agnostic learning under general distributions $\mathcal{D}$, we are forced to make some distributional assumptions. The main assumption in this paper is that $\mathcal{D}$ is a *product probability distribution* on $\mathcal{X}$; i.e., the n attributes are independent. For a discussion of this assumption and extensions, see Section 1.3.

1.2 When polynomial regression works

Although the SVM algorithm is very popular in practice, the scenarios in which it provably learns successfully are relatively few. Let us consider the SVM algorithm with degree-d polynomial kernel. The traditional SVM analysis is predicated on the assumption that the data is perfectly linearly separable in the polynomial feature space. Indeed, the heuristic arguments in support of good generalization are predicated on the data being separable *with large margin*. Even just the assumption of perfect separation may well be unreasonable. For example, suppose the target t is the very simple function given the by intersection of two homogeneous linear threshold functions over $\mathbb{R}^n$; i.e.,

$$t : \mathbb{R}^n \to \{-1, 1\}, \qquad t(x) = \mathrm{sgn}(w_1 \cdot x) \wedge \mathrm{sgn}(w_2 \cdot x).$$

It is known [MP69] that this target cannot be classified by the sign of a degree-d polynomial in the attributes for *any* finite d; this holds even when $n = 2$. Alternatively, when t is the intersection of two linear threshold functions over $\{0, 1\}^n$, it is not currently known if t can be classified by the sign of a degree-d polynomial for any $d < n - 1$. [OS03]

Because of this problem, one usually considers the "soft margin SVM algorithm" [CV95]. As mentioned, when this is run with no "regularization", the algorithm is essentially equivalent to degree-d polynomial regression with hinge loss. To show that this algorithm even has a chance of learning efficiently, one must be able to show that simple target functions can at least be *approximately* classified by the sign of low-degree polynomials. Of course, even stating any such result requires distributional assumptions. Let us make the following definition:

Definition 1.1 *Let $\mathcal{D}$ be a probability distribution on $\{0, 1\}^N$ and let $t : \{0, 1\}^N \to \mathbb{R}$. We say that t is* ϵ-concentrated up to degree d (under $\mathcal{D}$) *if there exists a polynomial $p : \{0, 1\}^N \to \mathbb{R}$ of degree at most d which has squared loss at most ϵ under $\mathcal{D}$; i.e., $\mathbf{E}_{\boldsymbol{x}\sim\mathcal{D}}[(p(\boldsymbol{x}) - t(\boldsymbol{x}))^2] \leq \epsilon$.*

It is well known that under the above conditions, $h := \mathrm{sgn}(p)$ has classification error at most ϵ under $\mathcal{D}$. Further, it is relatively easy to show that if $\mathcal{C}$ is a class of functions each of which is ϵ-concentrated up to degree d, then the degree-d polynomial regression algorithm with squared loss will PAC-learn $\mathcal{C}$ to accuracy $O(\epsilon)$ under $\mathcal{D}$.

The first result along these lines was due to Linial, Mansour, and Nisan [LMN93] who introduced the "Low-Degree Algorithm" for PAC-learning under the uniform distribution on $\{0, 1\}^n$. They showed that if $f : \{0, 1\}^n \to \{-1, 1\}$ is computed by a circuit of size s and depth c then it is ϵ-concentrated up to degree $(O(\log(s/\epsilon)))^c$ under the uniform distribution. Some generalizations of this result [FJS91, Hås01] are discussed in Section 4.

Another result using this idea was due to Klivans, O'Donnell, and Servedio [KOS04]. They introduced the "noise sensitivity method" for showing concentration results under the uniform distribution on $\{0, 1\}^n$. In particular, they showed that any $t : \{0, 1\}^n \to \{-1, 1\}$ expressible as a function of k linear threshold functions is ϵ-concentrated up to degree $O(k^2/\epsilon^2)$ under the uniform distribution.

These works obtained PAC learning guarantees for the the polynomial regression algorithm — i.e., guarantees only holding under the somewhat unrealistic assumption that $\mathrm{Opt} = 0$. A significant step towards handling noise was taken in [KKMS05]. Therein it was observed that low-degree L_2^2-approximability bounds imply L_1-approximability bounds (and hinge loss bounds), and further, such bounds imply that the polynomial regression algorithm works in the *agnostic* learning model. Specifically, their work contains the following theorem:

[3] In this paper, boldface denotes random variables.

[4] The definition of agnostic learning is sometimes taken to require error at most $\mathrm{Opt} + \epsilon$ with high probability, rather than in expectation. However this is known [KKMS05] to require almost negligible additional overhead.

Theorem 1.2 *([KKMS05]) Let $\mathcal{D}$ be a distribution on $\{0,1\}^N$ and let $\mathcal{C}$ be a class of functions $\{0,1\}^N \to \{-1,1\}$ each of which is ϵ^2-concentrated up to degree d under $\mathcal{D}$. Then the degree-d polynomial regression algorithm with L_1 loss (or hinge loss [Kal06]) uses $\mathrm{poly}(N^d/\epsilon)$ time and examples, and agnostically learns with respect to $\mathcal{C}$ under $\mathcal{D}$.*

Thus one gets agnostic learning algorithms under the uniform distribution on $\{0,1\}^n$ with respect to the class of AC^0 circuits (time $n^{\mathrm{polylog}(n/\epsilon)}$) and the class of functions of k thresholds (time $n^{O(k^2/\epsilon^4)}$) — note that the latter is polynomial time assuming k and ϵ are constants. Kalai et al. also obtained related results for agnostically learning with respect to single threshold functions under Gaussian and log-concave distributions on $\mathbb{R}^n$.

1.3 Overview of our learning results

We view the work of [KKMS05] as the first provable guarantee that one can learn interesting, broad classes of functions under the realistic noise model of agnostic learning (and in particular, that SVM-type methods can have this guarantee). One shortcoming of the present state of knowledge is that we have good concentration bounds for classes essentially only with respect to the uniform distribution on $\{0,1\}^n$ and the Gaussian distribution on $\mathbb{R}^n$.[5]

In this work we significantly broaden the class of distributions for which we can prove good concentration bounds, and hence for which we can prove the polynomial regression algorithm performs well. Roughly speaking, we show how to generalize any concentration result for the uniform distribution on $\{0,1\}^n$ into the same concentration result for *arbitrary product distributions* $\mathcal{D}$ on instance spaces $\mathcal{X} = X_1 \times \cdots \times X_n$.

We believe this is a significant generalization for several reasons. First, even just for the instance space $\{0,1\}^n$ the class of arbitrary product distributions is much more reasonable than the single distribution in which each attribute is 0 or 1 with probability exactly $1/2$. Our results are even stronger than this, though: they give on algorithm that works simultaneously for any product distribution over *any* instance space $\mathcal{X} = X_1 \times \cdots \times X_n$ where each $|X_i| \leq \mathrm{poly}(n)$.

Because we can handle non-binary attributes, the restriction to product spaces becomes much less severe. A common criticism of learning results under the uniform distribution or product distributions on $\{0,1\}^n$ is that they make the potentially unreasonable assumption that attributes are independent. However with our results, one can somewhat circumvent this. Suppose one believes that the attributes $X_1, \ldots, X_n$ are mostly independent, but some groups of them (e.g., height and weight) have mutual dependencies. One can then simply group together any dependent attribute sets $X_{i_1}, \ldots, X_{i_t}$ into a single "super-attribute" set $(X_{i_1} \times \cdots \times X_{i_t})$. Assuming that this eliminates dependencies — i.e., the new (super-)attributes are all independent — and that each $|X_{i_1} \times \cdots \times X_{i_t}|$ is still at most $\mathrm{poly}(n)$, one can proceed to use the polynomial regression algorithm. Here we see the usefulness of being able to handle arbitrary product distributions on arbitrary product sets.

In many reasonable cases our results can also tolerate the attribute sets X_i having superpolynomial size. What is really necessary is that the probability distribution on each X_i is mostly concentrated on polynomially many attributes. Indeed, we can further handle the common case when attributes are real-valued. As long as the probability distributions on real-valued attributes are not extremely skewed (e.g., Gaussian, exponential, Laplace, Pareto, chi-square, ...) our learning results go through after doing a naive "bucketing" scheme.

Finally, being able to learn under arbitrary product distributions opens the door to learning under *mixtures of product distributions*. Such mixtures — especially mixtures of Gaussians — are widely used as data distribution models in learning theory. We show that agnostic learning under mixtures can be reduced to agnostic learning under single product distributions. If the mixture distribution is precisely known to the algorithm, it can learn even under a mixture of polynomially many product distributions. Otherwise, when the mixture is unknown, we first need to use an algorithm for learning (or clustering) a mixture of product distributions from unlabeled examples. This is a difficult but well-studied problem. Using results of Feldman, O'Donnell, and Servedio [FOS05, FOS06] we can extend all of our agnostic learning results to learning under mixtures of constantly many product distributions with each $|X_i| \leq O(1)$ and constantly many (axis-aligned) Gaussian distributions.

1.4 Outline of technical results

In Section 2 we recall the orthogonal decomposition of functions on product spaces, as well as the more recently-studied notions of concentration and noise sensitivity on such spaces. In particular, we observe that if one can prove a good noise sensitivity bound for a class $\mathcal{C}$ under a product distribution Π, then [KKMS05] implies that the polynomial regression algorithm yields a good agnostic learner with respect to $\mathcal{C}$ under Π.

Section 3 contains the key reduction from noise sensitivity in general product spaces to noise sensitivity under the uniform distribution on $\{0,1\}^n$. It is carried out in the model case of linear threshold functions, which Peres [Per04] proved have δ-noise sensitivity at most $O(\sqrt{\delta})$. We give a surprisingly simple proof of the following:

Theorem 3.2 *Let $f : \mathcal{X} \to \{-1,1\}$ be a linear threshold function, where $\mathcal{X} = X_1 \times \cdots \times X_n$ has the product distribution $\Pi = \pi_1 \times \cdots \times \pi_n$. Then $\mathbb{NS}_\delta(f) \leq O(\sqrt{\delta})$.*

Proving this just in the case of a p-biased distribution on $\{0,1\}^n$ was an open problem suggested in [Per04]. This noise sensitivity bound thus gives us the following learning result:

[5] [FJS91] gives bounds for AC^0 under constant-bounded product distributions on $\{0,1\}^n$; [KKMS05] gives inexplicit bounds for a single threshold function under log-concave distributions on $\mathbb{R}^n$.

Theorem 3.4 *Let $\Pi = \pi_1 \times \cdots \times \pi_n$ be any product distribution over an instance space $\mathcal{X} = X_1 \times \cdots \times X_n$, where we assume $|X_i| \leq \mathrm{poly}(n)$ for each i. Let $\mathcal{C}$ denote the class of functions of k linear threshold functions over $\mathcal{X}$. Taking $d = O(k^2/\epsilon^4)$, the degree-d polynomial regression algorithm with L_1 loss (or hinge loss) uses $n^{O(k^2/\epsilon^4)}$ time and examples and agnostically learns with respect to $\mathcal{C}$.*

In Section 4 we discuss how to extend concentration results for other concept classes from uniform on $\{0,1\}^n$ to arbitrary product distributions on product spaces $\mathcal{X} = X_1 \times \cdots \times X_n$. Of course, it's not immediately clear, given a concept class $\mathcal{C}$ of functions on $\{0,1\}^n$, what it even means for it to be generalized to functions on $\mathcal{X}$. We discuss a reasonable such notion based on one-out-of-k encoding, and illustrate it in the case of AC^0 functions. The idea in this section is simple: any concentration result under uniform on $\{0,1\}^n$ easily implies a (slightly weaker) noise sensitivity bound; this can be translated into the same noise sensitivity bound under any product distribution using the methods of Section 3. In turn, that implies a concentration bound in the general product space. As an example, we prove the following:

Theorem 4.2 *Let $\mathcal{C}$ be the class of functions $X_1 \times \cdots \times X_n \to \{-1,1\}$ computed by unbounded fan-in circuit of size at most s and depth at most c (under the one-out-of-k encoding). Assume $|X_i| \leq \mathrm{poly}(n)$ for each i. Let Π be any product distribution on $X_1 \times \cdots \times X_n$. Then polynomial regression agnostically learns with respect to $\mathcal{C}$ under arbitrary product distributions in time $n^{(O(\log(s/\epsilon)))^{c-1}/\epsilon^2}$.*

Section 5 describes extensions of our learning algorithm to cases beyond those in which one has exactly a product distribution on an instance space $\mathcal{X} = X_1 \times \cdots \times X_n$ with each $|X_i| \leq \mathrm{poly}(n)$: these extensions include distributions "bounded by" or "close to" product distributions, as well as certain cases when the X_i's have superpolynomial cardinality or are $\mathbb{R}$. We end Section 5 with a discussion of learning under mixtures of product distributions. Here there is a distinction between learning when the mixture distribution is *known* to the algorithm and when it is *unknown*. In the former case we prove, e.g.:

Theorem 5.16 *Let $\mathcal{D}$ be any* known *mixture of $\mathrm{poly}(n)$ product distributions over an instance space $\mathcal{X} = X_1 \times \cdots \times X_n$, where we assume $|X_i| \leq \mathrm{poly}(n)$ for each i. Then there is a $n^{O(k^2/\epsilon^4)}$-time algorithm for agnostically learning with respect to the class of functions of k linear threshold functions over $\mathcal{X}$ under $\mathcal{D}$.*

In the latter case, by relying on algorithms for learning mixture distributions from unlabeled data, we prove:

Theorem 5.18 *Let $\mathcal{D}$ be any* unknown *mixture of $O(1)$ product distributions over an instance space $\mathcal{X} = X_1 \times \cdots \times X_n$, where we assume either: a) $|X_i| \leq O(1)$ for each i; or b) each $X_i = \mathbb{R}$ and each product distribution is a mixture of axis-aligned ($\mathrm{poly}(n)$-bounded) Gaussians. Then there is a $n^{O(k^2/\epsilon^4)}$-time algorithm for agnostically learning with respect to the class of functions of k linear threshold functions over $\mathcal{X}$ under $\mathcal{D}$.*

2 Product probability spaces

In this section we consider functions $f : \mathcal{X} \to \mathbb{R}$, where $\mathcal{X} = X_1 \times \cdots \times X_n$ is a product set. We will also assume $\mathcal{X}$ is endowed with some product probability distribution $\Pi = \pi_1 \times \cdots \times \pi_n$. All occurrences of $\mathbf{Pr}[\cdot]$ and $\mathbf{E}[\cdot]$ are with respect to this distribution unless otherwise noted, and we usually write $\boldsymbol{x}$ for a random element of $\mathcal{X}$ drawn from Π. For simplicity we assume that each set X_i is finite.[6] The vector space $L^2(\mathcal{X}, \Pi)$ of all functions $f : \mathcal{X} \to \mathbb{R}$ is viewed as an inner product space under the inner product $\langle f, g \rangle = \mathbf{E}[f(\boldsymbol{x})g(\boldsymbol{x})]$. We will also use the notation

$$\|f\|_2 = \sqrt{\langle f, f \rangle} = \sqrt{\mathbf{E}[f(\boldsymbol{x})^2]}.$$

2.1 Orthogonal decomposition

As each X_i is just an abstract set, there is not an inherent notion of a degree-d polynomial on $\mathcal{X}$. Ultimately the polynomial regression algorithm identifies $\mathcal{X}$ with a subset of $\{0,1\}^{|X_1|+\cdots+|X_n|}$ via the"one-out-of-k encoding" and works with polynomials over this space. However to prove concentration results, we need to take a more abstract approach and consider the "(Hoeffding) orthogonal decomposition" of functions on product spaces; see [vM47, Hoe48, KR82, Ste86]. In this section we recall this notion with our own notation.

Definition 2.1 *We say a function $f : X_1 \times \cdots \times X_n \to \mathbb{R}$ is a* simple function of order d *if it depends on at most d coordinates.*

Definition 2.2 *We say a function $f : X_1 \times \cdots \times X_n \to \mathbb{R}$ is a* function of order d *if it is a linear combination of simple functions of order d. The set of all such functions is a linear subspace of $L^2(\mathcal{X}, \Pi)$ and we denote it by $\mathcal{H}^{\leq d}(\mathcal{X}, \Pi)$.*

Definition 2.3 *We say a function $f : X_1 \times \cdots \times X_n \to \mathbb{R}$ is a* function of order exactly d *if it is a function of order d and it is orthogonal to all functions of order $d-1$; i.e., $\langle f, g \rangle = 0$ for all $g \in \mathcal{H}^{\leq d-1}(\mathcal{X}, \Pi)$. This is again a linear subspace of $L^2(\mathcal{X}, \Pi)$ and we denote it by $\mathcal{H}^{=d}(\mathcal{X}, \Pi)$.*

Proposition 2.4 *The space $L^2(\mathcal{X}, \Pi)$ is the orthogonal direct sum of the $\mathcal{H}^{=d}(\mathcal{X}, \Pi)$ spaces,*

$$L^2(\mathcal{X}, \Pi) = \bigoplus_{d=0}^{n} \mathcal{H}^{=d}(\mathcal{X}, \Pi).$$

Definition 2.5 *By virtue of the previous proposition, every function $f : X_1 \times \cdots \times X_n \to \mathbb{R}$ can be uniquely expressed as*

$$f = f^{=0} + f^{=1} + f^{=2} + \cdots + f^{=n},$$

where $f^{=d} : X_1 \times \cdots \times X_n \to \mathbb{R}$ denotes the projection of f into $\mathcal{H}^{=d}(\mathcal{X}, \Pi)$. We call $f^{=d}$ the order d part of f. *We will also write*

$$f^{\leq d} = f^{=0} + f^{=1} + f^{=2} + \cdots + f^{=d}.$$

[6] In fact, we will only need that each $L^2(X_i, \pi_i)$ has a countable basis.

In the sequel we will write simply $\mathcal{H}^{=d}$ in place of $\mathcal{H}^{=d}(\mathcal{X}, \Pi)$, etc. Although we will not need it, we recall a further refinement of this decomposition:

Definition 2.6 *For each $S \subseteq [n]$ we define $\mathcal{H}^{\leq S}$ to be the subspace consisting of all functions depending only on the coordinates in S. We define $\mathcal{H}^S$ to be the further subspace consisting of those functions in $\mathcal{H}^{\leq S}$ that are orthogonal to all functions in $\mathcal{H}^{\leq R}$ for each $R \subsetneq S$.*

Proposition 2.7 *The space $L^2(\mathcal{X}, \Pi)$ is the orthogonal direct sum of the $\mathcal{H}^S$ spaces, $L^2(\mathcal{X}, \Pi) = \bigoplus_{S \subseteq [n]} \mathcal{H}^S$. Hence every function $f : X_1 \times \cdots \times X_n \to \mathbb{R}$ can be uniquely expressed as $f = \sum_{S \subseteq [n]} f^S$, where $f^S : X_1 \times \cdots \times X_n \to \mathbb{R}$ denotes the projection of f into $\mathcal{H}^S$. Denoting also $f^{\leq S} = \sum_{R \subseteq S} f^R$ for the projection of f into $\mathcal{H}^{\leq S}$, we have the following interpretations:*

$$f^{\leq S}(y_1, \ldots, y_n) = \mathbf{E}[f(\boldsymbol{x}_1, \ldots, \boldsymbol{x}_n) \mid \boldsymbol{x}_i = y_i \; \forall \, i \in S];$$

$$f^S(x_1, \ldots, x_n) = \sum_{R \subseteq S} (-1)^{|S|-|R|} f^{\leq R}.$$

Finally, we connect the orthogonal decomposition of functions $f : \mathcal{X} \to \mathbb{R}$ with their analogue under the one-out-of-k encoding:

Proposition 2.8 *A function $f : \mathcal{X} \to \mathbb{R}$ is of order d if and only if its analogue $f : \{0,1\}^{|X_1|+\cdots+|X_n|} \to \mathbb{R}$ under the one-out-of-k encoding is expressible as a polynomial of degree at most d.*

2.2 Low-order concentration

As in the previous section we consider functions $f : \mathcal{X} \to \mathbb{R}$ under a product distribution Π. We will be especially interested in classifiers, functions $f : \mathcal{X} \to \{-1, 1\}$. Our goal is to understand and develop conditions under which such f can be approximated in squared loss by low-degree polynomials.

By basic linear algebra, we have the following:

Proposition 2.9 *Given $f : \mathcal{X} \to \mathbb{R}$, the best order-$d$ approximator to f under squared loss is $f^{\leq d}$. I.e.,*

$$\min_{g \text{ of order } d} \mathbf{E}[(f(\boldsymbol{x}) - g(\boldsymbol{x}))^2] = \|f - f^{\leq d}\|_2^2 = \sum_{i=d+1}^{n} \|f^{=i}\|_2^2.$$

Definition 2.10 *Given $f : \mathcal{X} \to \mathbb{R}$ we say that f is ϵ-concentrated up to order d if $\sum_{i=d+1}^{n} \|f^{=i}\|_2^2 \leq \epsilon$.*

By Proposition 2.8 we conclude the following:

Proposition 2.11 *Let $f : \mathcal{X} \to \mathbb{R}$ and identify f with a function $\{0,1\}^N \to \mathbb{R}$ under the one-out-of-k encoding. Then there exists a polynomial $p : \{0,1\}^N \to \mathbb{R}$ of degree at most d which ϵ-approximates f in squared loss under Π if and only if f is ϵ-concentrated up to order d.*

Combining this with the KKMS Theorem 1.2, we get the following learning result about polynomial regression:

Theorem 2.12 *Let $\Pi = \pi_1 \times \cdots \times \pi_n$ be a product distribution on $\mathcal{X} = X_1 \times \cdots \times X_n$. Write N for the total number of possible attribute values, $N = |X_1| + \cdots + |X_n|$. Let $\mathcal{C}$ be a class of functions $\mathcal{X} \to \{-1, 1\}$ each of which is ϵ^2-concentrated up to order d under Π. Then the degree-d polynomial regression algorithm with L_1 loss (or hinge loss) uses $\mathrm{poly}(N^d/\epsilon)$ time and examples, and agnostically learns with respect to $\mathcal{C}$ under Π.*

We will now show how to prove low-order concentration results by extending the "noise sensitivity method" of [KOS04] to general product spaces.

2.3 Noise sensitivity

We recall the generalization of noise sensitivity [BKS99] to general product spaces, described in [MOO05].

Definition 2.13 *Given $x \in X_1 \times \cdots \times X_n$ and $0 \leq \rho \leq 1$, we define a ρ-*noisy copy of *x to be a random variable $\boldsymbol{y}$ with distribution $N_\rho(x)$, where this denotes that each $\boldsymbol{y}_i$ is chosen to equal x_i with probability ρ and to be randomly drawn from π_i with probability $1 - \rho$, independently across i.*

Definition 2.14 *The* noise operator *T_ρ on functions $f : \mathcal{X} \to \mathbb{R}$ is given by*

$$(T_\rho f)(x) = \mathbf{E}_{\boldsymbol{y} \sim N_\rho(x)}[f(\boldsymbol{y})].$$

The noise stability *of f at ρ is*

$$\mathbb{S}_\rho(f) = \langle f, T_\rho f \rangle.$$

When $f : \mathcal{X} \to \{-1, 1\}$ we also define the noise sensitivity *of f at $\delta \in [0, 1]$ to be*

$$\mathbb{NS}_\delta(f) = \tfrac{1}{2} - \tfrac{1}{2}\mathbb{S}_{1-\delta}(f) = \mathop{\mathbf{Pr}}_{\substack{\boldsymbol{x} \sim \Pi \\ \boldsymbol{y} \sim \mathbb{N}_{1-\delta}(\boldsymbol{x})}} [f(\boldsymbol{x}) \neq f(\boldsymbol{y})].$$

The connection between noise stability, sensitivity, and concentration comes from the following two facts:

Proposition 2.15 *([MOO05]) For any $f : \mathcal{X} \to \mathbb{R}$,*

$$\mathbb{S}_\rho(f) = \sum_{i=0}^{n} \rho^i \|f^{=i}\|_2^2.$$

Proposition 2.16 *([KOS04]) Suppose $\mathbb{NS}_\delta(f) \leq \epsilon$. Then f is $\frac{2}{1-1/e}\epsilon$-concentrated up to order $1/\delta$.*

For example, Peres proved the following theorem:

Theorem 2.17 *([Per04]) If $f : \{0,1\}^n \to \{-1, 1\}$ is a linear threshold function then*

$$\mathbb{NS}_\delta(f) \leq O(1)\sqrt{\delta}$$

(under the uniform distribution on $\{0,1\}^n$). From [O'D03] we have that the $O(1)$ can be taken to be $\frac{5}{4}$ for every value of n and δ.

It clearly follows that if f is any function of k linear threshold functions then $\mathbb{NS}_\delta(f) \leq \frac{5}{4}k\sqrt{\delta}$. Combining this with Proposition 2.16:

Theorem 2.18 *([KOS04]) Let $f : \{0,1\}^n \to \{-1, 1\}$ be any function of k linear threshold functions. Then f is $(4k/\sqrt{d})$-concentrated up to order d under the uniform distribution, for any $d \geq 1$. In particular, f is ϵ^2-concentrated up to order $O(k^2/\epsilon^4)$.*

3 Noise sensitivity of threshold functions in product spaces

In this section we show that Peres's theorem can be extended to hold for linear threshold functions in all product spaces.

Definition 3.1 *We say a function $f : X_1 \times \cdots \times X_n \to \{-1,1\}$ is a* linear threshold function *if its analogue $f : \{0,1\}^N \to \{-1,1\}$ under one-out-of-k encoding is expressible as a linear threshold function. Equivalently, f is a linear threshold function if there exist weight functions $w_i : X_i \to \mathbb{R}$, $i = 1 \ldots n$, and a number $\theta \in \mathbb{R}$ such that*

$$f(x_1, \ldots, x_n) = \operatorname{sgn}\left(\sum_{i=1}^{n} w_i(x_i) - \theta\right).$$

No version of Peres's Theorem 2.17 was previously known to hold even in the simple case of linear threshold functions on $\{0,1\}^n$ under a p-biased product distribution with $p \neq 1/2$. Understanding just this nonsymmetric case was left as an open question in [Per04]. We now show that threshold functions over general product spaces are no more noise sensitive than threshold functions over $\{0,1\}^n$ under the uniform distribution.

Theorem 3.2 *Let $f : \mathcal{X} \to \{-1,1\}$ be a linear threshold function, where $\mathcal{X} = X_1 \times \cdots \times X_n$ has the product distribution $\Pi = \pi_1 \times \cdots \times \pi_n$. Then $\mathbb{NS}_\delta(f) \leq \frac{5}{4}\sqrt{\delta}$.*

Proof: For a pair of instances $z_0, z_1 \in \mathcal{X}$ and a vector $x \in \{0,1\}^n$, we introduce the notation z_x for the instance whose ith attribute $(z_x)_i$ is the ith attribute of z_{x_i}. For any fixed $z_0, z_1 \in \mathcal{X}$ we can define $g_{z_0,z_1} : \{0,1\}^n \to \{-1,1\}$ such that $g_{z_0,z_1}(x) = f(z_x)$. Note that this function is a linear threshold function in the traditional binary sense.

Let $\boldsymbol{z}_0, \boldsymbol{z}_1$ now denote independent random draws from Π, and let $\boldsymbol{x}$ denote a uniformly random vector from $\{0,1\}^n$. We have that $\boldsymbol{z}_{\boldsymbol{x}}$ is distributed as a random draw from Π. Further pick $\boldsymbol{y} \in \{0,1\}^n$ to be a δ-noisy copy of $\boldsymbol{x}$, i.e. $\boldsymbol{y} \sim N_\delta(\boldsymbol{x})$. Then $\boldsymbol{z}_{\boldsymbol{y}}$ is distributed as $N_\delta(\boldsymbol{z}_{\boldsymbol{x}})$. We now have

$$\begin{aligned}
\mathbb{NS}_\delta(f) &= \mathop{\mathbf{Pr}}_{\boldsymbol{z}_0,\boldsymbol{z}_1,\boldsymbol{x},\boldsymbol{y}}[f(\boldsymbol{z}_{\boldsymbol{x}}) \neq f(\boldsymbol{z}_{\boldsymbol{y}})] \\
&= \mathop{\mathbf{E}}_{\boldsymbol{z}_0,\boldsymbol{z}_1}\left[\mathop{\mathbf{Pr}}_{\boldsymbol{x},\boldsymbol{y}}[f(\boldsymbol{z}_{\boldsymbol{x}}) \neq f(\boldsymbol{z}_{\boldsymbol{y}})]\right] \\
&= \mathop{\mathbf{E}}_{\boldsymbol{z}_0,\boldsymbol{z}_1}\left[\mathop{\mathbf{Pr}}_{\boldsymbol{x},\boldsymbol{y}}[g_{\boldsymbol{z}_0,\boldsymbol{z}_1}(\boldsymbol{x}) \neq g_{\boldsymbol{z}_0,\boldsymbol{z}_1}(\boldsymbol{y})]\right].
\end{aligned}$$

Once z_0 and z_1 are fixed, the quantity in the expectation is just the noise sensitivity at δ of the binary linear threshold function $g_{\boldsymbol{z}_0,\boldsymbol{z}_1}$) which we can bound by $\frac{5}{4}\sqrt{\delta}$ using Theorem 2.17. So

$$\begin{aligned}
\mathbb{NS}_\delta(f) &= \mathop{\mathbf{E}}_{\boldsymbol{z}_0,\boldsymbol{z}_1}\left[\mathop{\mathbf{Pr}}_{\boldsymbol{x},\boldsymbol{y}}[g_{\boldsymbol{z}_0,\boldsymbol{z}_1}(\boldsymbol{x}) \neq g_{\boldsymbol{z}_0,\boldsymbol{z}_1}(\boldsymbol{y})]\right] \\
&\leq \mathop{\mathbf{E}}_{\boldsymbol{z}_0,\boldsymbol{z}_1}\left[\tfrac{5}{4}\sqrt{\delta}\right] = \tfrac{5}{4}\sqrt{\delta},
\end{aligned}$$

which is what we wanted to show. □

As with Theorem 2.18, we conclude:

Theorem 3.3 *Let $f : \mathcal{X} \to \{-1,1\}$ be any function of k linear threshold functions, where $\mathcal{X} = X_1 \times \cdots \times X_n$ has the product distribution $\Pi = \pi_1 \times \cdots \times \pi_n$. Then f is $(4k/\sqrt{d})$-concentrated up to order d, for any $d \geq 1$. In particular, f is ϵ^2-concentrated up to order $O(k^2/\epsilon^4)$.*

By combining Theorem 3.3 with our main learning theorem, Theorem 2.12, we conclude:

Theorem 3.4 *Let $\Pi = \pi_1 \times \cdots \times \pi_n$ be any product distribution over an instance space $\mathcal{X} = X_1 \times \cdots \times X_n$, where we assume $|X_i| \leq \mathrm{poly}(n)$ for each i. Let $\mathcal{C}$ denote the class of functions of k linear threshold functions over $\mathcal{X}$. Taking $d = O(k^2/\epsilon^4)$, the degree-d polynomial regression algorithm with L_1 loss (or hinge loss) uses $n^{O(k^2/\epsilon^4)}$ time and examples and agnostically learns with respect to $\mathcal{C}$.*

4 Concentration for other classes under product distributions

In this section we illustrate how essentially any result about ϵ-concentration of classes of functions under the uniform distribution on $\{0,1\}^n$ can be translated into a similar result for general product distributions. Besides linear threshold functions, the other main example of concentration comes from the original application of the Low Degree Algorithm [LMN93]: learning AC^0 functions in quasipolynomial time. Recall that AC^0 is the class of functions computed by unbounded fan-in circuits of constant depth and polynomial size. We will use this as a running example.

Suppose $\mathcal{C}$ is a class of functions $\mathcal{X} \to \{-1,1\}$, where $\mathcal{X} = X_1 \times \cdots \times X_n$. As usual, under the one-out-of-k encoding we can think of $\mathcal{C}$ as a class of functions $\{0,1\}^N \to \{-1,1\}$. In our example, this gives a reasonable notion of "AC^0 circuits on general product sets $\mathcal{X}$". Suppose further that $\overline{\mathcal{C}} \supseteq \mathcal{C}$ is any class of functions $\{0,1\}^N \to \{-1,1\}$ which is closed under negation of inputs and closed under fixing inputs to 0 or 1. In our example, the class of AC^0 circuits indeed has this basic property (as does the more precisely specified class of all circuits with size at most s and depth at most c).

Now by repeating the proof of Theorem 3.2, it is easy to see that any upper bound one can prove on the noise sensitivity of functions in $\overline{\mathcal{C}}$ under the uniform distribution on $\{0,1\}^N$ immediately translates an identical bound on the noise sensitivity of functions in $\mathcal{C}$ on $\mathcal{X}$ under any product distribution. The only thing to notice is that the functions $g_{\boldsymbol{z}_0,\boldsymbol{z}_1}$ arising in that proof will be in the class $\overline{\mathcal{C}}$. Thus we are reduced to proving noise sensitivity bounds for functions on $\{0,1\}^n$ under the uniform distribution.

Furthermore, any result on ϵ-concentration of functions on $\{0,1\}^n$ under the uniform distribution can be easily translated into a noise sensitivity bound which is not much worse:

Proposition 4.1 *Suppose that $f : \{0,1\}^n \to \{-1,1\}$ is ϵ-concentrated up to degree d under the uniform distribution on $\{0,1\}^n$. Then $\mathbb{NS}_{\epsilon/d}(f) \leq \epsilon$.*

Proof: Using traditional Fourier notation instead of orthogonal decomposition notation, we have

$$\mathbb{S}_{1-\epsilon/d}(f) = \sum_{S\subseteq[n]} (1-\epsilon/d)^{|S|}\hat{f}(S)^2$$
$$\geq (1-\epsilon/d)^d(1-\epsilon) \geq (1-\epsilon)^2,$$

where the first inequality used the fact that f is ϵ-concentrated up to degree d. Thus

$$\mathbb{NS}_{1-\epsilon/d}(f) = \tfrac{1}{2} - \tfrac{1}{2}\mathbb{S}_{1-\epsilon/d}(f) \leq \tfrac{1}{2} - \tfrac{1}{2}(1-\epsilon)^2 \leq \epsilon.$$

□

Finally, applying Proposition 2.16, we get $O(\epsilon)$-concentration up to order d/ϵ for the original class $\mathcal{C}$ of functions $\mathcal{X} \to \{-1,1\}$, under any product distribution on $\mathcal{X}$. This leads to an agnostic learning result for $\mathcal{C}$ under arbitrary product distributions which is the same as the one would get for $\mathcal{C}$ under the uniform distribution on $\{0,1\}^n$, except for an extra factor of ϵ in the running time's exponent.

For example, with regard to AC^0 functions, [LMN93, Hås01] proved the following:

Theorem 4.2 *Let $f : \{0,1\}^n \to \{-1,1\}$ be computable by an unbounded fan-in circuit of size at most s and depth at most c. Then f is ϵ-concentrated up to degree $d = (O(\log(s/\epsilon)))^{c-1}$.*

We therefore may conclude:

Theorem 4.3 *Let $\mathcal{C}$ be the class of functions $X_1 \times \cdots \times X_n \to \{-1,1\}$ computed by unbounded fan-in circuit of size at most s and depth at most c (under the one-out-of-k encoding). Assume $|X_i| \leq \mathrm{poly}(n)$ for each i. Let Π be any product distribution on $X_1 \times \cdots \times X_n$. Then every $f \in \mathcal{C}$ is $\frac{2}{1-1/e}\epsilon$-concentrated up to order $d = (O(\log(s/\epsilon)))^{c-1}/\epsilon$. As a consequence, polynomial regression agnostically learns with respect to $\mathcal{C}$ under arbitrary product distributions in time $n^{(O(\log(s/\epsilon)))^{c-1}/\epsilon^2}$.*

This result should be compared to the following theorem from Furst, Jackson, and Smith [FJS91] for PAC-learning under bounded product distributions on $\{0,1\}^n$:

Theorem 4.4 ([FJS91]) *The class $\mathcal{C}$ of functions $\{0,1\}^n \to \{-1,1\}$ computed by unbounded fan-in circuit of size at most s and depth at most c can be PAC-learned under any product distribution in time $n^{(O((1/p)\log(s/\epsilon)))^{c+O(1)}}$, assuming the mean of each coordinate is in the range $[p, 1-p]$.*

The advantage of the result from [FJS91] is that it does not pay the extra $1/\epsilon^2$ in the exponent. The advantages of our result is that it holds under arbitrary product distributions on product sets. (Our result is in the agnostic model, but the result of [FJS91] could also be by applying the results of [KKMS05].)

5 Extensions

5.1 Distributions close to or dominated by product distributions

We begin with some simple observations showing that the underlying distribution need not be *precisely* a product distribution. First, the following fact can be considered standard:

Proposition 5.1 *Suppose that under distribution $\mathcal{D}$, algorithm $\mathcal{A}$ agnostically learns with respect to class $\mathcal{C}$, using m examples to achieve error ϵ. If $\mathcal{D}'$ is any distribution satisfying $\|\mathcal{D}' - \mathcal{D}\|_1 \leq \epsilon/m$, then $\mathcal{A}$ also agnostically learns under $\mathcal{D}'$, using m examples to achieve error $2\epsilon + 2\epsilon/m \leq 4\epsilon$.*

Proof: The key fact we use is that if $\boldsymbol{X}$ is a random variable with $|\boldsymbol{X}| \leq 1$ always, then $|\mathbf{E}_{\mathcal{D}'}[\boldsymbol{X}] - \mathbf{E}_{\mathcal{D}}[\boldsymbol{X}]| \leq \|\mathcal{D}' - \mathcal{D}\|_1$. This implies that for any hypothesis h, $|\mathrm{err}_{\mathcal{D}'}(h) - \mathrm{err}_{\mathcal{D}}(h)| \leq \epsilon/m$. In particular, it follows that $\mathrm{Opt}_{\mathcal{D}'} \leq \mathrm{Opt}_{\mathcal{D}} + \epsilon/m$. Further, let $\boldsymbol{h}$ be the random variable denoting the hypothesis $\mathcal{A}$ produces when given examples from $\mathcal{D}^{\otimes m}$. By assumption, we have

$$\mathop{\mathbf{E}}_{\mathcal{D}^{\otimes m}}[\mathrm{err}_{\mathcal{D}}(\boldsymbol{h})] \leq \mathrm{Opt}_{\mathcal{D}} + \epsilon$$

which is at most $\mathrm{Opt}_{\mathcal{D}'} + \epsilon + \epsilon/m$. Since $\|\mathcal{D}'^{\otimes m} - \mathcal{D}^{\otimes m}\|_1 \leq m(\epsilon/m) = \epsilon$, the key fact applied to $\mathrm{err}_{\mathcal{D}}(\boldsymbol{h})$ implies

$$\mathop{\mathbf{E}}_{\mathcal{D}'^{\otimes m}}[\mathrm{err}_{\mathcal{D}}(\boldsymbol{h})] \leq \mathrm{Opt}_{\mathcal{D}'} + \epsilon + \epsilon/m + \epsilon.$$

Finally, as we saw, $\mathrm{err}_{\mathcal{D}'}(\boldsymbol{h}) \leq \mathrm{err}_{\mathcal{D}}(\boldsymbol{h}) + \epsilon/m$ always. Thus

$$\mathop{\mathbf{E}}_{\mathcal{D}'^{\otimes m}}[\mathrm{err}_{\mathcal{D}'}(\boldsymbol{h})] \leq \mathrm{Opt}_{\mathcal{D}'} + 2\epsilon + 2\epsilon/m,$$

completing the proof. □

We will use the above result later when learning under mixtures of product distributions.

A simple extension to the case when the distribution is "dominated" by a product distribution was already pointed out in [KKMS05]:

Observation 5.2 *Let $\mathcal{D}$ be a distribution on $\mathcal{X}$ which is "C-dominated" by a product probability distribution $\Pi = \pi_1 \times \cdots \times \pi_n$; i.e., for all $x \in \mathcal{X}$, $\mathcal{D}(x) \leq C\Pi(x)$. If f is ϵ-concentrated up to degree d under Π, then f is $C\epsilon$-concentrated up to degree d under $\mathcal{D}$.*

Hence:

Theorem 5.3 *Suppose we are in the setting of Theorem 3.4 except that Π is any distribution which is C-dominated by a product probability distribution. Then the degree-d polynomial regression algorithm learns with respect to $\mathcal{C}$ with $d = O(C^2k^2/\epsilon^4)$ and hence $n^{O(C^2k^2/\epsilon^4)}$ time and examples.*

5.2 Larger attribute domains

So far we have assumed that each attribute space X_i is only of polynomial cardinality. This can fairly easily be relaxed to the assumption that most of the probability mass in each (X_i, π_i) is concentrated on polynomially many atoms. Let us begin with some basic preliminaries:

Notation 5.4 *Given a distribution π on a set X, as well as a subset $X' \subseteq X$, we use the notation π' for the distribution on X' given by conditioning π on this set. (We always assume $\pi(X') \neq 0$.)*

Fact 5.5 *Let $\mathcal{X} = X_1 \times \cdots \times X_n$ and let $\Pi = \pi_1 \times \cdots \times \pi_n$ be a product distribution on $\mathcal{X}$. Let $X_i' \subseteq X_i$, $i = 1 \ldots n$, and write Π' for the distribution Π conditioned on the set $\mathcal{X}' = X_1' \times \cdots \times X_n'$. Then Π' is the product distribution $\pi_1' \times \cdots \times \pi_n'$.*

We now observe that if $\mathcal{X}'$ is a "large" subset of $\mathcal{X}$, then any two functions which are close in $L^2(\mathcal{X}, \Pi)$ are also close in $L^2(\mathcal{X}', \Pi')$:

Proposition 5.6 *In the setting of Fact 5.5, suppose that $\Pr_{\boldsymbol{x}_i \sim \pi_i}[\boldsymbol{x}_i \notin X_i'] \leq 1/(2n)$ for all i. Then for any two functions $f : \mathcal{X} \to \mathbb{R}$ and $g : \mathcal{X} \to \mathbb{R}$,*

$$\|f|_{\mathcal{X}'} - g|_{\mathcal{X}'}\|_{2,\Pi'}^2 \leq 2 \cdot \|f - g\|_{2,\Pi}^2$$

where $f|_{\mathcal{X}'} : \mathcal{X}' \to \mathbb{R}$ denotes the restriction of f to $\mathcal{X}'$, and similarly for $g|_{\mathcal{X}'}$.

Proof: Writing $h = f - g$, we have

$$\begin{aligned} \|h\|_{2,\Pi}^2 &= \mathop{\mathbf{E}}_{\boldsymbol{x} \sim \Pi}[h(\boldsymbol{x})^2] \\ &= \mathop{\mathbf{Pr}}_{\boldsymbol{x} \sim \Pi}[\boldsymbol{x} \in \mathcal{X}'] \cdot \mathop{\mathbf{E}}_{\boldsymbol{x} \sim \Pi}[h(\boldsymbol{x})^2 \mid \boldsymbol{x} \in \mathcal{X}'] \\ &\quad + \mathop{\mathbf{Pr}}_{\boldsymbol{x} \sim \Pi}[\boldsymbol{x} \notin \mathcal{X}'] \cdot \mathop{\mathbf{E}}_{\boldsymbol{x} \sim \Pi}[h(x)^2 \mid \boldsymbol{x} \notin \mathcal{X}']. \end{aligned}$$

Using $\mathbf{E}_{\boldsymbol{x} \sim \Pi}[h(\boldsymbol{x}) \mid \boldsymbol{x} \notin \mathcal{X}'] \geq 0$, we have

$$\begin{aligned} \|h\|_{2,\Pi}^2 &\geq \mathop{\mathbf{Pr}}_{\boldsymbol{x} \sim \Pi}[\boldsymbol{x} \in \mathcal{X}'] \cdot \mathop{\mathbf{E}}_{\boldsymbol{x} \sim \Pi}[h(x)^2 \mid \boldsymbol{x} \in \mathcal{X}'] \\ &= \mathop{\mathbf{Pr}}_{\boldsymbol{x} \sim \Pi}[\boldsymbol{x} \in \mathcal{X}'] \cdot \mathop{\mathbf{E}}_{\boldsymbol{x} \sim \Pi'}[h(x)^2]. \end{aligned}$$

But by the union bound

$$\mathop{\mathbf{Pr}}_{\boldsymbol{x} \sim \Pi}[\boldsymbol{x} \notin \mathcal{X}'] \leq \sum_{i=1}^{n} \mathop{\mathbf{Pr}}_{\boldsymbol{x}_i \sim \Pi_i}[\boldsymbol{x}_i \notin X_i'] \leq n \cdot 1/(2n) = 1/2,$$

so $\Pr_{\boldsymbol{x} \sim \Pi}[\boldsymbol{x} \in \mathcal{X}'] \geq 1/2$. Thus

$$2 \cdot \|h\|_{2,\Pi}^2 \geq \mathop{\mathbf{E}}_{\boldsymbol{x} \sim \Pi'}[h(x)^2] = \|f|_{\mathcal{X}'} - g|_{\mathcal{X}'}\|_{2,\Pi'}^2,$$

completing the proof. □

Corollary 5.7 *In the setting of the previous proposition, if f is ϵ-concentrated up to order d under Π, then $f|_{\mathcal{X}'}$ is 2ϵ-concentrated up to order d under Π'.*

Proof: It suffices to observe that if $g : \mathcal{X} \to \mathbb{R}$ is a function of order d, then $g|_{\mathcal{X}'}$ is also a function of order d. □

We can now describe an extended learning algorithm which works when the attribute spaces are mostly supported on sets of polynomial cardinality:

Definition 5.8 *We say that a finite probability space (X, π) is* (η, r)-bounded *if there exists a subset $X' \subseteq X$ of cardinality at most $|X'| \leq r$ such that $\Pr_{\boldsymbol{x} \sim \pi}[\boldsymbol{x} \notin X'] \leq \eta$.*

Our algorithm will learn whenever all n attribute sets are, say, $(\epsilon/n, \mathrm{poly}(n))$-bounded. The first step of the algorithm will be to determine a set of attribute values which contain almost all of the probability mass.

Lemma 5.9 *Let (X, π) be an (η, r)-bounded probability space. Let $\boldsymbol{\mathcal{Z}}$ be a set of $m = r \ln(r/\delta)/\eta$ samples drawn independently from π. Define Y to be the set $\{x \in X : x$ was sampled in $\boldsymbol{\mathcal{Z}}\}$. Then with probability at least $1 - \delta$, the set Y satisfies $\mathbf{Pr}_{\boldsymbol{x} \sim \pi}[\boldsymbol{x} \notin Y] \leq 2\eta$.*

Proof: In fact, we will prove the slightly stronger statement that with probability at least $1 - \delta$ the set Y satisfies $\Pr_{\boldsymbol{x} \sim \pi}[\boldsymbol{x} \notin Y \cap X'] \leq 2\eta$, where X' is any set fulfilling the (η, r)-boundedness condition of (X, π).

To prove the claim, we split the sampling procedure into r epochs, where we draw $\ln(r/\delta)/\eta$ samples in each epoch. Let Y_i be the set of all atoms in X sampled among the first i epochs, with Y_0 denoting the empty set. We will prove that with probability at least $1 - \delta$, the following holds for all epochs $i \in [r]$: either Y_{i-1} satisfies $\Pr_{\boldsymbol{x} \sim \pi}[\boldsymbol{x} \notin Y_{i-1} \cap X'] \leq 2\eta$, or $(Y_i \cap X') \setminus Y_{i-1} \neq \emptyset$ (i.e., we see a "new" atom from X' in the ith epoch).

Let's first note that satisfying the above conditions implies that in the end $\Pr_{\boldsymbol{x} \sim \pi}[\boldsymbol{x} \notin Y \cap X'] \leq 2\eta$. This is straightforward: if at any epoch Y_{i-1} satisfies $\Pr_{\boldsymbol{x} \sim \pi}[\boldsymbol{x} \notin Y_{i-1} \cap X'] \leq 2\eta$ then we're done because $Y \supseteq Y_{i-1}$. Otherwise, in all r epochs we see a new atom from X', and hence at the end of the sampling we will have seen r distinct atoms of X'; then $|X'| \leq r$ implies that our final $Y \supseteq X'$.

Now to complete the proof let us bound the probability that for a given $i \in [r]$ the Y_{i-1} does not satisfy $\Pr_{\boldsymbol{x} \sim \pi}[\boldsymbol{x} \notin Y_{i-1} \cap X'] \leq 2\eta$ and we do not see a new element of X' in the ith epoch. Note that if $\Pr_{\boldsymbol{x} \sim \pi}[\boldsymbol{x} \notin Y_{i-1} \cap X'] > 2\eta$, then the fact that $\Pr_{\boldsymbol{x} \sim \pi}[\boldsymbol{x} \notin X'] \leq \eta$ implies that $\Pr_{\boldsymbol{x} \sim \pi}[\boldsymbol{x} \in X' \setminus Y_{i-1}] > \eta$. So the probability that we do not observe any element of $X' \setminus Y_{i-1}$ in $\ln(r/\delta)/\eta$ samples is

$$\begin{aligned} \mathop{\mathbf{Pr}}_{\boldsymbol{x} \sim \pi}[x \notin X' \setminus Y_{i-1}]^{\ln(r/\delta)/\eta} &\leq (1 - \eta)^{\ln(r/\delta)/\eta} \\ &\leq e^{-\eta \cdot \ln(r/\delta)/\eta} = \delta/r. \end{aligned}$$

By applying the union bound, we see that there is probability at most δ that any of the r epochs fails, so we're done. □

We now give our extended learning algorithm:

1. Draw a set $\boldsymbol{\mathcal{Z}}_1$ of m_1 unlabeled examples.
2. Draw a set $\boldsymbol{\mathcal{Z}}_2$ of m_2 labeled examples.
3. Delete from $\boldsymbol{\mathcal{Z}}_2$ any instance/label pair where the instance contains an attribute value not appearing in $\boldsymbol{\mathcal{Z}}_1$.
4. Run the degree-d polynomial regression algorithm on $\boldsymbol{\mathcal{Z}}_2$.

Theorem 5.10 *Let $\Pi = \pi_1 \times \cdots \times \pi_n$ be a product distribution on the set $\mathcal{X} = X_1 \times \cdots \times X_n$ and assume each probability space (X_i, π_i) is $(\epsilon/n, r)$-bounded. Write $N = nr$. Let $\mathcal{C}$ be a class of functions $\mathcal{X} \to \{-1, 1\}$ each of which is ϵ^2-concentrated up to order d. Set $m_1 = \mathrm{poly}(N/\epsilon)$ and $m_2 = \mathrm{poly}(N^d/\epsilon)$. The above algorithm uses $\mathrm{poly}(N^d/\epsilon)$ time and examples and agnostically learns with respect to $\mathcal{C}$ under Π.*

Proof: For simplicity we will equivalently prove that the algorithm outputs a hypothesis with error at most $\mathrm{Opt} + O(\epsilon)$, rather than $\mathrm{Opt} + \epsilon$.

We first want to establish that with probability at least $1 - \epsilon$, the set of attributes observed in the sample $\boldsymbol{\mathcal{Z}}_1$ covers almost all of the probability mass of Π. For each $i \in [n]$, let X_i' be the set of attribute values from X_i observed in at least one of the samples in $\boldsymbol{\mathcal{Z}}_1$. Using the fact that each (X_i, π_i) is $(\epsilon/n, r)$-bounded, Lemma 5.9 implies that for sufficiently large $m_1 = \mathrm{poly}(N/\epsilon)\log(1/\epsilon)$, each X_i' will satisfy $\Pr_{\boldsymbol{x}_i \sim \pi_i}[\boldsymbol{x}_i \notin X_i'] \leq 2\epsilon/n$ except with probability at most ϵ/n. Applying the union bound, all X_i' simultaneously satisfy the condition with probability at least $1-\epsilon$. We henceforth assume this happens. Writing $\mathcal{X}' = X_1' \times \cdots \times X_n'$, we note that, by the union bound, $\Pr_{\boldsymbol{x} \sim \Pi}[\boldsymbol{x} \notin \mathcal{X}'] \leq 2\epsilon$.

The second thing we establish is that we do not throw away too many examples in Step 3 of the algorithm. We have just observed that the probability a given example in $\boldsymbol{\mathcal{Z}}_2$ is deleted is at most 2ϵ. We may assume $2\epsilon \leq 1/2$, and then a Chernoff bound (and $m_2 \gg \log(1/\epsilon)$) easily implies that with probability at least $1 - \epsilon$, at most, say, two-thirds of all examples are deleted. Assuming this happens, we have that even after deletion, $\boldsymbol{\mathcal{Z}}_2$ still contains at least $\mathrm{poly}(N^d/\epsilon)$ many examples.

We now come to the main part of the proof, which is based on the observation that the undeleted examples in $\boldsymbol{\mathcal{Z}}_2$ are distributed as i.i.d. draws from the restricted product distribution Π' gotten by conditioning Π on $\mathcal{X}'$. Thus we are in a position to apply our main learning result, Theorem 2.12. The polynomial regression part of the above algorithm indeed uses $\mathrm{poly}(N^d/\epsilon)$ time and examples, and it remains to analyze the error of the hypothesis it outputs.

First, we use the fact that each function f in $\mathcal{C}$ is ϵ^2-concentrated up to order d to conclude that each function $f|_{\mathcal{X}'}$ in "$\mathcal{C}|_{\mathcal{X}'}$" is $2\epsilon^2$-concentrated up to order d. This uses Proposition 5.6 and the fact that we may assume $2\epsilon \leq 1/2$. Next, the guarantee of Theorem 2.12 is that when learning the target classifier t (viewed as a function $\mathcal{X} \to \{-1, 1\}$ or $\mathcal{X}' \to \{-1, 1\}$), the expected error under Π' of the hypothesis h produced is at most $\mathrm{Opt}' + O(\epsilon)$, where

$$\mathrm{Opt}' = \min_{f' \in \mathcal{C}|_{\mathcal{X}'}} \mathop{\mathbf{Pr}}_{\boldsymbol{x} \sim \Pi'}[f'(\boldsymbol{x}) \neq t(\boldsymbol{x})].$$

By definition, there is a function $f \in \mathcal{C}$ satisfying

$$\mathop{\mathbf{Pr}}_{\boldsymbol{x} \sim \Pi}[f(\boldsymbol{x}) \neq t(\boldsymbol{x})] = \mathrm{Opt}.$$

Since $\Pr_{\boldsymbol{x} \sim \Pi}[\boldsymbol{x} \notin \mathcal{X}'] \leq 2\epsilon$, it is easy to see that $f|_{\mathcal{X}'}$ has error at most $\mathrm{Opt} + 2\epsilon$ on t under Π'. Thus $\mathrm{Opt}' \leq \mathrm{Opt} + 2\epsilon$, and we conclude that the expected error under Π' of h on t is at most $\mathrm{Opt} + 2\epsilon + O(\epsilon) = \mathrm{Opt} + O(\epsilon)$. Finally, the same observation implies that the expected error under Π of h on t is at most $\mathrm{Opt} + 2\epsilon + O(\epsilon) = \mathrm{Opt} + O(\epsilon)$.

We have thus established that with probability at least $1 - 2\epsilon$, the polynomial regression part of the above algorithm outputs a hypothesis with expected error at most $\mathrm{Opt} + O(\epsilon)$. It follows that the overall expected error is at most $\mathrm{Opt} + O(\epsilon)$, as desired. □

5.3 Real-valued attributes

We next consider the particular case of learning with respect to linear threshold functions, but when some of the attributes are *real-valued*. This case is relatively easily handled by discretizing the ranges of the distributions and using the previously discussed techniques. Our approach works for a very wide variety of distributions on $\mathbb{R}$; these distributions need not even be continuous. We only need the distributions to satisfy "polynomial boundedness and anti-concentration" bounds.

Definition 5.11 *We say that a distribution $\mathcal{D}$ over $\mathbb{R}$ is B-*polynomially bounded *if for all $\eta > 0$, there is an interval I of length at most $\mathrm{poly}(B/\eta)$ such that $\Pr_{\boldsymbol{x} \sim \mathcal{D}}[\boldsymbol{x} \notin I] \leq \eta$.*

Definition 5.12 *Given a real-valued random variable $\boldsymbol{x}$ with distribution $\mathcal{D}$, recall that the* Lévy (anti-)concentration function *$Q(\boldsymbol{x}; \lambda)$ is defined by*

$$Q(\boldsymbol{x}; \lambda) = \sup_{t \in \mathbb{R}} \mathop{\mathbf{Pr}}_{\boldsymbol{x} \sim \mathcal{D}}[\boldsymbol{x} \in [t - \lambda/2, t + \lambda/2]].$$

*We say that $\mathcal{D}$ has B-*polynomial anti-concentration *if $Q(\mathcal{D}; \lambda) \leq \mathrm{poly}(B) \cdot \lambda^c$ for some positive $c > 0$. Note that if $\mathcal{D}$ is a continuous distribution with pdf everywhere at most B then it has B-polynomial anti-concentration (with $c = 1$ in fact).*

Having polynomial boundedness and concentration is an extremely mild condition; for example, the following familiar continuous distributions are all B-polynomial bounded and have B-polynomial anti-concentration: *Gaussians* with $1/B \leq \sigma^2 \leq B$; *exponential* distributions with $1/B \leq \lambda \leq B$; *Laplace* distributions with scale parameter with $1/B \leq b \leq B$; *Pareto* distributions with shape parameter $1/B \leq k \leq B$; *chi-square* distributions with variance $1/B \leq \sigma^2 \leq B$ (for 1 degree of freedom, the anti-concentration "c" needs to be $1/2$); etc.

(Furthermore, in most cases even the condition on the parameter being in $[1/B, B]$ can be eliminated. For example, suppose the first coordinate has a Gaussian distribution with standard deviation σ. With $O(\log(1/\delta))$ examples, one can with probability at least $1 - \delta$ estimate σ to within a multiplicative factor of 2. Having done so, one can multiply all examples' first coordinate by an appropriate constant so as to get a Gaussian distribution with standard deviation in $[1/2, 2]$. Further, this does not change the underlying agnostic learning problem, since the class of linear threshold functions is closed under scaling a coordinate. For clarity of exposition, we leave further considerations of this sort to the

reader.)

We now describe the effect that discretizing a real-valued distribution can have with respect to functions of linear threshold functions. It is convenient to switch from working with a distribution on $\mathcal{X}$ and target function $\mathcal{X} \to \{-1,1\}$ to just having a distribution $\mathcal{D}$ on $\mathcal{X} \times \{-1,1\}$ — see the discussion after definition of agnostic learning in Section 1.1. As usual, assume that $\mathcal{X} = X_1 \times \cdots \times X_n$ is a product set and that the marginal distribution of $\mathcal{D}$ on $\mathcal{X}$ is a product distribution.

Suppose we have one coordinate with a real-valued distribution; without loss of generality, say $X_1 = \mathbb{R}$, and write $\mathcal{D}_1$ for the marginal distribution of $\mathcal{D}$ on X_1. When we refer to a "linear threshold function" on $\mathcal{X}$, we assume that the "weight function" $w_1 : X_1 \to \mathbb{R}$ for coordinate 1 is just $w_1(x_1) = c_1 x_1$ for some nonzero constant c_1.

Lemma 5.13 *Let $\mathcal{C}$ denote the class of functions of k linear threshold functions over $\mathcal{X}$. As usual, write*

$$\mathrm{Opt} = \inf_{f \in \mathcal{C}} \mathrm{err}_{\mathcal{D}}(f), \qquad \textit{where } \mathrm{err}_{\mathcal{D}}(f) = \mathop{\mathbf{Pr}}_{(\boldsymbol{x},y)\sim\mathcal{D}}[f(\boldsymbol{x}) \neq \boldsymbol{y}].$$

Consider discretizing $X_1 = \mathbb{R}$ by mapping each $x_1 \in \mathbb{R}$ to $\mathrm{rd}_\tau(x_1)$, the nearest integer multiple of τ to x_i. Write $X_1' = \tau\mathbb{Z}$ and let $\mathcal{D}'$ denote the distribution on $X_1' \times X_2 \times \cdots X_n \times \{-1,1\}$ induced from $\mathcal{D}$ by the discretization.[7] Write Opt' for the quantity analogous to Opt for $\mathcal{D}'$. Then if $\mathcal{D}_1$ has B-polynomial anti-concentration, it holds that $\mathrm{Opt}' \leq \mathrm{Opt} + k \cdot \mathrm{poly}(B) \cdot \tau^{\Omega(1)}$.

Proof: It suffices to show that for any $f \in \mathcal{C}$,

$$k \cdot \mathrm{poly}(B) \cdot \tau^{\Omega(1)} \geq |\mathrm{err}_{\mathcal{D}}(f) - \mathrm{err}_{\mathcal{D}'}(f)| = \left| \mathop{\mathbf{Pr}}_{(\boldsymbol{x},\boldsymbol{y})\sim\mathcal{D}}[f(\boldsymbol{x}) \neq \boldsymbol{y}] - \mathop{\mathbf{Pr}}_{(\boldsymbol{x},\boldsymbol{y})\sim\mathcal{D}'}[f(\boldsymbol{x}) \neq \boldsymbol{y}] \right|.$$

Writing Π for the marginal of $\mathcal{D}$ on $\mathcal{X}$, we can prove the above by proving

$$\mathop{\mathbf{Pr}}_{\boldsymbol{x}\sim\Pi}[f(\boldsymbol{x}) \neq f(\mathrm{rd}_\tau(\boldsymbol{x}_1), \boldsymbol{x}_2, \ldots, \boldsymbol{x}_n)] \leq k \cdot \mathrm{poly}(B) \cdot \tau^{\Omega(1)}.$$

Since f is a function of some k linear threshold functions, by the union bound it suffices to show

$$\mathop{\mathbf{Pr}}_{\boldsymbol{x}\sim\Pi}[h(\boldsymbol{x}) \neq h(\mathrm{rd}_\tau(\boldsymbol{x}_1), \boldsymbol{x}_2, \ldots, \boldsymbol{x}_n)] \leq \mathrm{poly}(B) \cdot \tau^{\Omega(1)}$$

for any linear threshold function h. We can do this by showing

$$\mathop{\mathbf{Pr}}_{\substack{\boldsymbol{x}_1\sim\mathcal{D}_1 \\ \boldsymbol{Y}}}[\mathrm{sgn}(c_1\boldsymbol{x}_1 + \boldsymbol{Y}) \neq \mathrm{sgn}(c_1 \mathrm{rd}_\tau(\boldsymbol{x}_1) + \boldsymbol{Y})] \leq \mathrm{poly}(B) \cdot \tau^{\Omega(1)},$$

where $\boldsymbol{Y}$ is the random variable distributed according to the other part of the linear threshold function h. Note that $\boldsymbol{Y}$ and $\boldsymbol{x}_1$ are independent because Π is a product distribution. Now since $|\boldsymbol{x}_1 - \mathrm{rd}_\tau(\boldsymbol{x}_1)|$ is always at most $\tau/2$, we can only have $\mathrm{sgn}(c_1\boldsymbol{x}_1 + \boldsymbol{Y}) \neq \mathrm{sgn}(c_1 \mathrm{rd}_\tau(\boldsymbol{x}_1) + \boldsymbol{Y})$ if

$$|c_1\boldsymbol{x}_1 + \boldsymbol{Y}| \leq |c_1|\tau/2 \quad \Leftrightarrow \quad |\boldsymbol{x}_1 + \boldsymbol{Y}/c_1| \leq \tau/2.$$

[7] This can lead to inconsistent labels, which is why we switched to $\mathcal{D}$ rather than have a target function.

It is an easy and well-known fact that if $\boldsymbol{x}$ and $\boldsymbol{y}$ are independent random variables then $Q(\boldsymbol{x}+\boldsymbol{y};\lambda) \leq Q(\boldsymbol{x};\lambda)$; hence

$$\mathop{\mathbf{Pr}}_{\substack{\boldsymbol{x}_1\sim\mathcal{D}_1 \\ \boldsymbol{Y}}}[|\boldsymbol{x}_1 + \boldsymbol{Y}/c_1| \leq \tau/2] \leq Q(\boldsymbol{x}_1; \tau/2).$$

But $\mathcal{D}_1$ has B-polynomial anti-concentration, so $Q(\boldsymbol{x}_1; \tau/t) \leq \mathrm{poly}(B) \cdot \tau^{\Omega(1)}$, as needed. □

By repeating this lemma up to n times, it follows that even if all n coordinate distributions are real-valued, so long as they have $\mathrm{poly}(n)$-polynomial anti-concentration we will suffer little error. Specifically (assuming $k \leq \mathrm{poly}(n)$ as well), by taking $\tau = \mathrm{poly}(\epsilon/n)$ we get that discretization only leads to an additional error of ϵ.

Finally, note that if a distribution $\mathcal{D}_i$ is $\mathrm{poly}(n)$-polynomially bounded then its discretized version is $(\epsilon/n, \mathrm{poly}(n/\epsilon))$-bounded in the sense of Section 5.2; this lets us apply Theorem 5.10. Summarizing:

Theorem 5.14 *Let $\Pi = \pi_1 \times \cdots \times \pi_n$ be a product distribution on the set $\mathcal{X} = X_1 \times \cdots \times X_n$. For the finite X_i's, assume each is $(\epsilon/n, \mathrm{poly}(n/\epsilon))$-bounded. For the real X_i's, assume the associated π_i is $\mathrm{poly}(n)$-polynomially bounded and has $\mathrm{poly}(n)$-polynomial anti-concentration. Let $\mathcal{C}$ denote the class of functions of at most $k \leq \mathrm{poly}(n)$ linear threshold functions over $\mathcal{X}$. Then there is a $\mathrm{poly}(n/\epsilon)^{k^2/\epsilon^4}$ time algorithm which agnostically learns with respect to $\mathcal{C}$ under Π.*

5.4 Mixtures of product distributions

So far we have only considered learning under distributions $\mathcal{D}$ that are product distributions. In this section we show how to handle the commonly-studied case of mixtures of product distributions.

The first step is to show a generic learning-theoretic reduction: Roughly speaking, if we can agnostically learn with respect to any one of a family of distributions, then we can agnostically learn with respect to a *known* mixture of distributions from this family — even a mixture of polynomially many such distributions. (In our application the family of distributions will be the product distributions, but our reduction does not rely on this.) Although the following theorem uses relatively standard ideas, we do not know if it has appeared previously in the literature:

Theorem 5.15 *Let $\mathfrak{D}$ be a family of distributions over an instance space $\mathcal{X}$. There is a generic reduction from the problem of agnostically learning under a* known *mixture of c distributions from $\mathfrak{D}$ to the problem of agnostically learning under a single known distribution from $\mathfrak{D}$. The reduction incurs a running time slowdown of $\mathrm{poly}(cT)/\gamma$ for an additional error of γ, where T denotes the maximum time needed to compute $\mathcal{D}(x)$ for a mixture component $\mathcal{D}$.*

Proof: Suppose we are agnostically learning (with respect to some class $\mathcal{C}$) under the distribution $\mathcal{D}$ which is a mixture of c distributions $\mathcal{D}_1, \ldots, \mathcal{D}_c$ with mixing weights $p_1, \ldots, p_c$. We make the assumption that the learning algorithm knows each of the mixing weights p_i, each of the distributions $\mathcal{D}_i$,

and can compute any of the probabilities $\mathcal{D}_i(x)$ in time T. We assume in the following that the $\mathcal{D}_i$'s are discrete distributions, but the case of absolutely continuous distributions could be treated in essentially the same way.

First, we claim that the algorithm can simulate learning under any of the single distributions $\mathcal{D}_i$, with slowdown $\mathrm{poly}(cT)/p_i$. This is a standard proof based on rejection sampling: given an example x, the algorithm retains it with probability

$$r_i(x) := p_i \frac{\mathcal{D}_i(x)}{\mathcal{D}(x)}, \tag{1}$$

a quantity the algorithm can compute in time $\mathrm{poly}(cT)$. One can check that this leads to the correct distribution $\mathcal{D}_i$ on instances. The probability of retaining an example is easy seen to be precisely $1/p_i$, leading to the stated slowdown.

The main part of the proof now involves showing that if the algorithm agnostically learns under each $\mathcal{D}_i$, it can combine the hypotheses produced into an overall hypothesis which is good under $\mathcal{D}$. We will deal with the issue of running time (in particular, very small p_i's) at the end of the proof. Let Opt denote the minimal error achievable among functions in $\mathcal{C}$ under $\mathcal{D}$, and write Opt_i for the analogous quantity under $\mathcal{D}_i$, $i = 1 \ldots c$. Since one could use the same $f \in \mathcal{C}$ for each $\mathcal{D}_i$, clearly $\mathrm{Opt} \geq \sum_{i=1}^{c} p_i \mathrm{Opt}_i$. By reduction, the algorithm produces hypotheses $\boldsymbol{h}_1, \ldots, \boldsymbol{h}_c$ satisfying $\mathbf{E}[\mathrm{err}_{\mathcal{D}_i}(\boldsymbol{h}_i)] \leq \mathrm{Opt}_i + \epsilon$.

We allow our overall algorithm to output a *randomized* hypothesis $\boldsymbol{h}$. We will then show that $\mathbf{E}[\mathrm{err}_{\mathcal{D}}(\boldsymbol{h})] \leq \mathrm{Opt} + \epsilon$. where the expectation is over the subalgorithms' production of the $\boldsymbol{h}_i$'s plus the "internal coins" of $\boldsymbol{h}$. Having shown this, it follows that our algorithm could equally well produce a deterministic hypothesis, just by (randomly) fixing a setting of $\boldsymbol{h}$'s internal coins as its last step.

Assume for a moment that the subalgorithms' hypotheses are fixed, $h_1, \ldots, h_c$. The randomized overall hypothesis $\boldsymbol{h} : \mathcal{X} \to \{-1, 1\}$ is defined by taking $\boldsymbol{h}(x) = h_i(x)$ with probability exactly $r_i(x)$, where the probabilities $r_i(x)$ are as defined in (1). (Note that they indeed sum to 1 and are computable in time $\mathrm{poly}(cT)$.) Writing t for the target function, we compute:

$$\begin{aligned}
&\mathop{\mathbf{E}}_{\boldsymbol{h}\text{'s coins}}[\mathrm{err}_{\mathcal{D}}(\boldsymbol{h})] \\
&= \mathop{\mathbf{E}}_{\boldsymbol{x}\sim\mathcal{D}}[\mathop{\mathbf{Pr}}_{\boldsymbol{h}\text{'s coins}}[\boldsymbol{h}(\boldsymbol{x}) \neq t(\boldsymbol{x})]] \\
&= \mathop{\mathbf{E}}_{\boldsymbol{x}\sim\mathcal{D}}\left[\sum_{i:h_i(\boldsymbol{x})\neq t(\boldsymbol{x})} r_i(\boldsymbol{x})\right] \\
&= \mathop{\mathbf{E}}_{\boldsymbol{x}\sim\mathcal{D}}\left[\sum_{i:h_i(\boldsymbol{x})\neq t(\boldsymbol{x})} p_i(\boldsymbol{x})\frac{\mathcal{D}_i(\boldsymbol{x})}{\mathcal{D}(\boldsymbol{x})}\right] \\
&= \sum_{x\in\mathcal{X}} \sum_{i:h_i(x)\neq t(x)} p_i(x)\mathcal{D}_i(x) \\
&= \sum_{i=1}^{c} p_i \sum_{x:h_i(x)\neq t(x)} \mathcal{D}_i(x) = \sum_{i=1}^{c} p_i \mathrm{err}_{\mathcal{D}_i}(h_i).
\end{aligned}$$

We now take the expectation over the production of the subhypotheses and conclude

$$\begin{aligned}
\mathop{\mathbf{E}}_{\boldsymbol{h}}[\mathrm{err}_{\mathcal{D}}(\boldsymbol{h})] &= \sum_{i=1}^{c} p_i \mathbf{E}[\mathrm{err}_{\mathcal{D}_i}(\boldsymbol{h}_i)] \leq \sum_{i=1}^{c} p_i(\mathrm{Opt}_i + \epsilon) \\
&= \sum_{i=1}^{c} p_i \mathrm{Opt}_i + \epsilon \leq \mathrm{Opt} + \epsilon, \quad (2)
\end{aligned}$$

as claimed.

It remains to deal with small p_i's and analyze the running time slowdown. We modify the overall algorithm so that it only simulates and learns under $\mathcal{D}_i$ if $p_i \geq \gamma/c$. Thus the simulation slowdown we incur is only $\mathrm{poly}(cT)/\gamma$, as desired. For any i with $p_i < \gamma/c$ we use an arbitrary hypothesis h_i in the above analysis and assume only $\mathrm{err}_{\mathcal{D}_i}(h_i) \leq 1$. It is easy to see that this incurs an additional error in (2) of at most $\sum_{i:p_i<\gamma/c} p_i \leq \gamma$, as necessary. □

Combining Theorem 5.15 with, say, Theorem 3.4 (for simplicity), we may conclude:

Theorem 5.16 *Let $\mathcal{D}$ be any* known *mixture of $\mathrm{poly}(n)$ product distributions over an instance space $\mathcal{X} = X_1 \times \cdots \times X_n$, where we assume $|X_i| \leq \mathrm{poly}(n)$ for each i. Then there is a $n^{O(k^2/\epsilon^4)}$-time algorithm for agnostically learning with respect to the class of functions of k linear threshold functions over $\mathcal{X}$ under $\mathcal{D}$.*

When the mixture of product distributions is not known a priori, we can first run an algorithm for learning mixtures of product distributions from unlabeled examples. For example, Feldman, O'Donnell, and Servedio [FOS05] proved the following:

Theorem 5.17 ([FOS05]) *Let $\mathcal{D}$ be an unknown mixture of $c = O(1)$ many product distributions over an instance space $\mathcal{X} = X_1 \times \cdots \times X_n$, where we assume $|X_i| \leq O(1)$ for each i. There is an algorithm which, given i.i.d. examples from $\mathcal{D}$ and $\eta > 0$, runs in time $\mathrm{poly}(n/\eta)\log(1/\delta)$ and with probability at least $1 - \delta$ outputs the parameters of a mixture of c product distributions $\mathcal{D}'$ satisfying $\|\mathcal{D}' - \mathcal{D}\|_1 \leq \eta$.*

(The theorem was originally stated in terms of KL-divergence but also holds with L_1-distance [FOS05].) In [FOS06] the same authors gave an analogous result for the case when each $X_i = \mathbb{R}$ and each product distribution is a product of Gaussians with means and variances in $[1/\mathrm{poly}(n), \mathrm{poly}(n)]$.

We conclude:

Theorem 5.18 *Let $\mathcal{D}$ be any* unknown *mixture of $O(1)$ product distributions over an instance space $\mathcal{X} = X_1 \times \cdots \times X_n$, where we assume either: a) $|X_i| \leq O(1)$ for each i; or b) each $X_i = \mathbb{R}$ and each product distribution is a mixture of axis-aligned ($\mathrm{poly}(n)$-bounded) Gaussians. Then there is a $n^{O(k^2/\epsilon^4)}$-time algorithm for agnostically learning with respect to the class of functions of k linear threshold functions over $\mathcal{X}$ under $\mathcal{D}$.*

Proof: First use the results of [FOS05, FOS06] with $\eta = \epsilon/n^{O(k^2/\epsilon^4)}$, producing a known mixture distribution $\mathcal{D}'$ with $\|\mathcal{D}' - \mathcal{D}\|_1 \leq \epsilon/n^{O(k^2/\epsilon^4)}$. Then run the algorithm from Theorem 5.18. The conclusion now follows from Proposition 5.1. □

6 Conclusions

In this work, we have shown how to perform agnostic learning under arbitrary product distributions and even under limited mixtures of product distributions. The main technique was showing that noise sensitivity bounds under the uniform distribution on $\{0,1\}^n$ yield the same noise sensitivity bounds under arbitrary product distributions. The running time and examples required by our algorithm are virtually the same as those required for learning under the uniform distribution on $\{0,1\}^n$.

While we have established many interesting scenarios for which polynomial regression works, there is still significant room for extension. One direction is to seek out new concept classes and/or distributions for which polynomial regression achieves polynomial-time agnostic learning. Our work has dealt mostly in the case where all the attributes are mutually independent; it would be especially interesting to get learning under discrete distributions that are far removed from this assumption.

References

[BKS99] Itai Benjamini, Gil Kalai, and Oded Schramm. Noise sensitivity of Boolean functions and applications to percolation. *Publ. Math. de l'IHÉS*, 90(1):5–43, 1999.

[CV95] Corinna Cortes and Vladimir Vapnik. Support-vector networks. *Machine Learning*, 20(3):273–297, 1995.

[FJS91] Merrick Furst, Jeffrey Jackson, and Sean Smith. Improved learning of AC^0 functions. In *Proc. 4th Workshop on Comp. Learning Theory*, pages 317–325, 1991.

[FOS05] Jonathan Feldman, Ryan O'Donnell, and Rocco Servedio. Learning mixtures of product distributions over discrete domains. In *Proc. 46th IEEE Symp. on Foundations of Comp. Sci.*, pages 501–510, 2005.

[FOS06] Jonathan Feldman, Ryan O'Donnell, and Rocco Servedio. Pac learning mixtures of gaussians with no separation assumption. In *Proc. 19th Workshop on Comp. Learning Theory*, pages 20–34, 2006.

[GR06] Venkatesan Guruswami and Prasad Raghavendra. Hardness of learning halfspaces with noise. In *Proc. 47th IEEE Symp. on Foundations of Comp. Sci.*, pages 543–552, 2006.

[Hås01] J. Håstad. A slight sharpening of LMN. *J. of Computing and Sys. Sci.*, 63(3):498–508, 2001.

[Hoe48] Wassily Hoeffding. A class of statistics with asymptotically normal distribution. *Ann. Math. Stat.*, 19(3):293–325, 1948.

[Kal06] Adam Kalai. Machine learning theory course notes. http://www.cc.gatech.edu/∼atk/teaching/mlt06/lectures/mlt-06-10.pdf, 2006.

[KKMS05] Adam Kalai, Adam Klivans, Yishay Mansour, and Rocco Servedio. Agnostically learning halfspaces. In *Proc. 46th IEEE Symp. on Foundations of Comp. Sci.*, pages 11–20, 2005.

[KOS04] Adam Klivans, Ryan O'Donnell, and Rocco Servedio. Learning intersections and thresholds of halfspaces. *J. of Computing and Sys. Sci.*, 68(4):808–840, 2004.

[KR82] Samuel Karlin and Yosef Rinott. Applications of Anova type decompositions for comparisons of conditional variance statistics including jackknife estimates. *Ann. Stat.*, 10(2):485–501, 1982.

[KSS94] Michael Kearns, Robert Schapire, and Linda Sellie. Toward efficient agnostic learning. *Machine Learning*, 17(2):115–141, 1994.

[LBW95] Wee Sun Lee, Peter Bartlett, and Robert Williamson. On efficient agnostic learning of linear combinations of basis functions. In *Proc. 8th Workshop on Comp. Learning Theory*, pages 369–376, 1995.

[LMN93] Nathan Linial, Yishay Mansour, and Noam Nisan. Constant depth circuits, Fourier transform, and learnability. *Journal of the ACM*, 40(3):607–620, 1993.

[MOO05] Elchanan Mossel, Ryan O'Donnell, and Krzysztof Oleszkiewicz. Noise stability of functions with low influences: invariance and optimality. In *Proc. 46th IEEE Symp. on Foundations of Comp. Sci.*, pages 21–30, 2005.

[MP69] Marvin Minsky and Symour Papert. *Perceptrons*. MIT Press, 1969.

[O'D03] Ryan O'Donnell. *Computational aspects of noise sensitivity*. PhD thesis, MIT, 2003.

[OS03] Ryan O'Donnell and Rocco Servedio. New degree bounds for polynomial threshold functions. In *Proc. 35th ACM Symp. on the Theory of Computing*, pages 325–334, 2003.

[Per04] Y. Peres. Noise stability of weighted majority. `arXiv:math/0412377v1`, 2004.

[Ste86] J. Michael Steele. An Efron-Stein inequality for nonsymmetric statistics. *Ann. Stat.*, 14(2):753–758, 1986.

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

[vM47] Richard von Mises. On the asymptotic distribution of differentiable statistical functions. *Ann. Math. Stat.*, 18(3):309–348, 1947.