E0 370 Statistical Learning Theory **Lecture 12 (Sep 27, 2011)**

Learnability: More Examples and Results

Lecturer: Shivani Agarwal *Scribe: Thirumulanathan D*

1 Introduction

In the last lecture we saw basic definitions of learnability, together with some initial examples and results. This lecture provides more detailed examples and results that will help to give a better understanding of both the possibilities and limitations of learnability.

2 Sufficient Conditions for Learnability and Upper Bounds on Sample Complexity

2.1 Learnability of a Binary-Valued Function Class

Recall from last time the following result, re-stated here in a more convenient form:

Theorem 2.1. Let $\mathcal{H} \subseteq \{\pm 1\}^{\mathcal{X}}$, with $\text{VCdim}(\mathcal{H}) = d < \infty$. Then:

(1) (General/agnostic setting) $\mathcal{H}$ is learnable by any algorithm $\mathcal{A} : \cup_{m=1}^{\infty}(\mathcal{X} \times \{\pm 1\})^m \to \mathcal{H}$ which, given a training sample $S \in (\mathcal{X} \times \{\pm 1\})^m$, outputs a function $h_S \in \arg\min_{h \in \mathcal{H}}(\text{er}_S[h])$ (i.e. any ERM algorithm in $\mathcal{H}$). Such an algorithm has sample complexity

$$m_{\mathcal{A}}(\epsilon, \delta) = O\left(\frac{1}{\epsilon^2}\left(d \ln\left(\frac{1}{\epsilon}\right) + \ln\left(\frac{1}{\delta}\right)\right)\right) . \tag{1}$$

If there exists such an algorithm that has running time $\text{poly}(m)$, then $\mathcal{H}$ is *efficiently* learnable.

(2) (Target function setting) $\mathcal{H}$ is learnable in the target function setting by any algorithm $\mathcal{A} : \cup_{t \in \mathcal{H}} \cup_{m=1}^{\infty} (\mathcal{X} \times t)^m \to \mathcal{H}$ which, given a training sample $S \in (\mathcal{X} \times t)^m$ for any $t \in \mathcal{H}$, outputs a function $h_S \in \mathcal{H}$ with $\text{er}_S[h_S] = 0$ (i.e. zero training error). Such an algorithm has sample complexity

$$m_{\mathcal{A}}(\epsilon, \delta) = O\left(\frac{1}{\epsilon}\left(d \ln\left(\frac{1}{\epsilon}\right) + \ln\left(\frac{1}{\delta}\right)\right)\right) . \tag{2}$$

If there exists such an algorithm that has running time $\text{poly}(m)$, then $\mathcal{H}$ is *efficiently* learnable in the target function setting.

Note: If $\mathcal{H}$ is finite, we can replace the $d \ln(\frac{1}{\epsilon})$ terms in the sample complexity expressions above by $\ln|\mathcal{H}|$; this may yield a small improvement in some cases (in other cases, the VC-dimension term is smaller).

Exercise. Let $\mathcal{H} \subseteq \{\pm 1\}^{\mathcal{X}}$ be finite. Show that for any distribution D on $\mathcal{X} \times \{\pm 1\}$ and any $m \in \mathbb{N}$, $\epsilon > 0$,

$$P_{S \sim D^m}\left(\exists h \in \mathcal{H} : \text{er}_S[h] = 0, \text{er}_D[h] \geq \epsilon\right) \leq |\mathcal{H}|\, e^{-m\epsilon}$$

(Hint: Show that the above probability is at most $|\mathcal{H}|\,(1-\epsilon)^m$.)

The following example reviews and builds on an example we saw last time:

Example 1 (Axis-parallel rectangles in the plane)**.** Let $\mathcal{X} = \mathbb{R}^2$, and let

$$\mathcal{H} = \{h : \mathcal{X} \to \{\pm 1\} \mid h(\mathbf{x}) = 1 \text{ if } \mathbf{x} \in [a,b] \times [c,d] \text{ and } -1 \text{ otherwise, for some } a \le b,\, c \le d\}\,.$$

We know $\text{VCdim}(\mathcal{H}) = 4$. Therefore, by Theorem 2.1:

(2) $\mathcal{H}$ is learnable in the target function setting by any algorithm that gives a function in $\mathcal{H}$ with zero training error. In particular, consider an algorithm which given $S \in (\mathcal{X} \times t)^m$ for some $t \in \mathcal{H}$, returns a function $h_S \in \mathcal{H}$ corresponding to the smallest rectangle enclosing all positive points in S (and some rectangle outside the training points if there are no positive examples in S). This requires computing the minimum and maximum coordinates among all positive points in S along each of the two dimensions, which can clearly be done in $O(m)$ time. Thus $\mathcal{H}$ is *efficiently learnable in the target function setting.*

(1) $\mathcal{H}$ is learnable (in the general/agnostic setting) by any ERM algorithm in $\mathcal{H}$. In particular, consider an algorithm which given $S \in (\mathcal{X} \times \{\pm 1\})^m$, sorts the points in ascending order along each dimension, and evaluates $\text{er}_S[h_{ijkl}]$ for all $1 \le i < j \le m$, $1 \le k < l \le m$, where h_{ijkl} corresponds to the rectangle defined by $[x_{i1}^{(1)}, x_{j1}^{(1)}] \times [x_{k2}^{(2)}, x_{l2}^{(2)}]$ with $\mathbf{x}_i^{(r)} \in \mathbb{R}^2$ being the point with the i^{th} largest coordinate in S in the r^{th} dimension, and returns a function h_{ijkl} with the smallest training error (or a rectangle outside S if this has smaller error). Clearly, this can be done in $O(m^4)$ time. Thus $\mathcal{H}$ is *efficiently learnable in the general/agnostic setting.*

2.2 Learnability w.r.t. Domain Dimension

The following result follows directly from Theorem 2.1 and definitions of learnability w.r.t. domain dimension:

Theorem 2.2. Let $\mathcal{H} = \cup_{n=1}^{\infty} \mathcal{H}_n$, where $\mathcal{H}_n \subseteq \{\pm 1\}^{\mathcal{X}_n}$ and instances in $\mathcal{X}_n$ can be encoded using a representation of size polynomial in n (typically, $\mathcal{X}_n \subseteq \mathbb{R}^n$). Let $\text{VCdim}(\mathcal{H}_n)$ be finite for all $n \in \mathbb{N}$. Then:

(1) (General/agnostic setting) The dimension-graded class $\mathcal{H} = \cup_{n=1}^{\infty} \mathcal{H}_n$ is learnable by any algorithm $\mathcal{A} : \cup_{n=1}^{\infty} \cup_{m=1}^{\infty} (\mathcal{X}_n \times \{\pm 1\})^m \to \mathcal{H}$ which given a training sample $S \in (\mathcal{X}_n \times \{\pm 1\})^m$, outputs a function $h_S \in \arg\min_{h \in \mathcal{H}_n}(\text{er}_S[h])$. Such an algorithm has sample complexity

$$m_{\mathcal{A}}(\epsilon, \delta, n) = O\left(\frac{1}{\epsilon^2}\left(\text{VCdim}(\mathcal{H}_n) \ln\left(\frac{1}{\epsilon}\right) + \ln\left(\frac{1}{\delta}\right)\right)\right)\,. \tag{3}$$

If $\text{VCdim}(\mathcal{H}_n) = \text{poly}(n)$ and there exists an algorithm as above that has running time $\text{poly}(m, n)$, then $\mathcal{H}$ is *efficiently* learnable.

(2) (Target function setting) The dimension-graded class $\mathcal{H} = \cup_{n=1}^{\infty} \mathcal{H}_n$ is learnable in the target function setting by any algorithm $\mathcal{A} : \cup_{n=1}^{\infty} \cup_{t \in \mathcal{H}_n} \cup_{m=1}^{\infty} (\mathcal{X}_n \times t)^m \to \mathcal{H}$ which given a training sample $S \in (\mathcal{X}_n \times t)^m$, outputs a function $h_S \in \mathcal{H}_n$ with $\text{er}_S[h_S] = 0$. Such an algorithm has sample complexity

$$m_{\mathcal{A}}(\epsilon, \delta, n) = O\left(\frac{1}{\epsilon}\left(\text{VCdim}(\mathcal{H}_n) \ln\left(\frac{1}{\epsilon}\right) + \ln\left(\frac{1}{\delta}\right)\right)\right)\,. \tag{4}$$

If $\text{VCdim}(\mathcal{H}_n) = \text{poly}(n)$ and there exists an algorithm as above that has running time $\text{poly}(m, n)$, then $\mathcal{H}$ is *efficiently* learnable in the target function setting.

Example 2 (Axis-parallel hyper-rectangles in $\mathbb{R}^n$)**.** Let $\mathcal{X} = \mathbb{R}^n$, and let

$$\mathcal{H}_n = \{h : \mathcal{X}_n \to \{\pm 1\} \mid h(\mathbf{x}) = 1 \text{ if } \mathbf{x} \in \Pi_{i=1}^{n}[a_i, b_i] \text{ and } -1 \text{ otherwise, for some } a_i \le b_i, i = 1, \ldots, n\}\,.$$

It can be shown that $\text{VCdim}(\mathcal{H}_n)$=2n. Therefore, by Theorem 2.2:

(2) $\mathcal{H} = \cup_{n=1}^{\infty} \mathcal{H}_n$ is learnable in the target function setting by any algorithm that given $S \in (\mathcal{X}_n \times t)^m$ returns $h_S \in \mathcal{H}_n$ with $\text{er}_S[h_S] = 0$. In particular, the algorithm given above for learning rectangles in $\mathbb{R}^2$ can be extended to compute the minimum and the maximum coordinates of positive points in S in each of n dimensions; this gives $O(mn)$ running time. Thus $\mathcal{H}$ is *efficiently learnable in the target function setting.*

(1) $\mathcal{H} = \cup_{n=1}^{\infty}\mathcal{H}_n$ is learnable (in the general/agnostic setting) by any algorithm that given $S \in (\mathcal{X}_n \times \{\pm 1\})^m$, outputs a function $h_S \in \arg\min_{h\in\mathcal{H}_n}(\mathrm{er}_S[h])$. In particular, the algorithm given above for learning rectangles in $\mathbb{R}^2$ in the general setting can be extended to give such an algorithm; however the running time in this case is $O(m^{2n})$. Therefore $\mathcal{H}$ is *learnable in the general/agnostic setting*; however this algorithm does not establish efficient learnability in the general setting.

Exercise. Give a poly(m, n) time algorithm to find a hyper-rectangle in $\mathbb{R}^n$ with minimal training error on a sample $S \in (\mathbb{R}^n \times \{\pm 1\})^m$, or show the problem is NP-hard.

Example 3 (k-DNF over $\{0,1\}^n$)**.** Let $\mathcal{X}_n = \{0,1\}^n$ and let $k \in \mathbb{N}$ be a constant. Let

$$\mathcal{H}_n = \{h : \mathcal{X}_n \to \{\pm 1\} \mid h(\mathbf{x}) = 1 \text{ if } c_{i_1}(\mathbf{x}) \vee c_{i_2}(\mathbf{x}) \vee \ldots \vee c_{i_r}(\mathbf{x}) = 1 \text{ and } -1 \text{ otherwise,} \\ \text{where } r \in \mathbb{N} \text{ and each } c_i(\mathbf{x}) \text{ is a conjunction of at most } k \text{ literals}\}.$$

Then as discussed last time, $|\mathcal{H}_n| \le 2^{(2n)^{k+1}}$, which gives $\mathrm{VCdim}(\mathcal{H}_n) \le \log_2|\mathcal{H}_n| \le (2n)^{k+1}$. Therefore, by Theorem 2.2:

(2) $\mathcal{H} = \cup_{n=1}^{\infty}\mathcal{H}_n$ is learnable in the target function setting. In particular, consider an algorithm which, given $X \in (\mathcal{X}_n \times t)^m$ for some $t \in \mathcal{H}_n$, starts with a k-DNF forumla that consists of a disjunction containing all possible terms (conjunctions) of at most k literals; for each negative example in S, removes all terms that are satisfied by the example; and then returns the remaining k-DNF formula. Clearly, the resulting function classifies all examples in S correctly; moreover, the running time of this algorithm is $O(m(2n)^{k+1})$. Thus $\mathcal{H}$ is *efficiently learnable in the target function setting.*

(1) $\mathcal{H} = \cup_{n=1}^{\infty}\mathcal{H}_n$ is *learnable in the general/agnostic setting.* Again, we leave it as an exercise to determine whether there is a polynomial time algorithm for finding a minimal training error k-DNF for a given training sample in the general setting.

Example 4 (k-CNF over $\{0,1\}^n$)**.** This is similar to the k-DNF example above; the details are left as an exercise.

Example 5 (Linear classifiers in $\mathbb{R}^n$)**.** Let $\mathcal{X}_n = \mathbb{R}^n$, and let

$$\mathcal{H}_n = \{h : \mathcal{X}_n \to \{\pm 1\} \mid h(\mathbf{x}) = \mathrm{sign}(\mathbf{w}\cdot\mathbf{x} + b) \text{ for some } \mathbf{w} \in \mathbb{R}^n,\, b \in \mathbb{R}\}.$$

We know $\mathrm{VCdim}(\mathcal{H}_n) = n + 1$. Therefore, by Theorem 2.2:

(2) $\mathcal{H} = \cup_{n=1}^{\infty}\mathcal{H}_n$ is learnable in the target function setting. In particular, one can use a polynomial-time algorithm based on linear programming (or the hard margin SVM algorithm) to find a zero training error linear classifier. Thus $\mathcal{H}$ is *efficiently learnable in the target function setting.*

(1) $\mathcal{H} = \cup_{n=1}^{\infty}\mathcal{H}_n$ is *learnable in the general/agnostic setting.* However ERM in this case is NP-hard, and so unless P = NP, there is no polynomial-time algorithm for finding a minimal training error linear classifier in the general setting.

2.3 Learnability w.r.t. Target Complexity and Occam's Razor

The following result provides a sufficient condition for learnability w.r.t. target complexity; a proof can be found in [2].

Theorem 2.3 (Blumer et al., 1989)**.** Let $\mathcal{H} \subseteq \{\pm 1\}^{\mathcal{X}}$, with target complexity function size : $\mathcal{H} \to \mathbb{N}$, and for $s \in \mathbb{N}$, let $\mathcal{H}_s = \{h \in \mathcal{H} \mid \mathrm{size}(h) \le s\}$. Then the complexity-graded class $\mathcal{H} = \cup_{s=1}^{\infty}\mathcal{H}_s$ is learnable (w.r.t. the complexity function 'size') by any algorithm $\mathcal{A}$ which given a training sample $S \in (\mathcal{X} \times t)^m$ for some $t \in \mathcal{H}_s$, returns as output a function $h_S \in \mathcal{H}^{\mathcal{A}}_{s,m} \subseteq \mathcal{H}$ such that $\mathrm{er}_S[h_S] = 0$ and $\mathrm{VCdim}(\mathcal{H}^{\mathcal{A}}_{s,m}) \le s^q m^\alpha$ for some $q \in \mathbb{N}$, $0 \le \alpha < 1$. Such an algorithm (often called an *Occam* algorithm) has sample complexity

$$m_{\mathcal{A}}(\epsilon, \delta, s) = O\left(\max\left(\left(\frac{s^q}{\epsilon}\ln\left(\frac{1}{\epsilon}\right)\right)^{\frac{1}{1-\alpha}}, \frac{1}{\epsilon}\ln\left(\frac{1}{\delta}\right)\right)\right). \tag{5}$$

If there exists such an algorithm with running time poly(m), then $\mathcal{H}$ is *efficiently* learnable w.r.t. the target complexity function 'size'.

Example 6. Let $\mathcal{X} = \mathbb{R}$, and let

$$\mathcal{H} = \{h : \mathcal{X} \to \{\pm 1\} \mid h(x) = 1 \text{ if } x \in \cup_{i=1}^{s}[a_i, b_i] \text{ and } -1 \text{ otherwise, for some } s \in \mathbb{N}, a_i \leq b_i\}.$$

Note that $\text{VCdim}(\mathcal{H}) = \infty$. Let $\text{size} : \mathcal{H} \to \mathbb{N}$ be defined as

$$\text{size}(h) = \min\{s \in \mathbb{N} \mid h \text{ can be written as a union of } s \text{ closed intervals}\}.$$

Now suppose $S \in (\mathcal{X} \times t)^m$ for some $t \in \mathcal{H}_s$. Consider an algorithm that given S as input, sorts the m points $x_1, x_2, \ldots, x_m$ in S in increasing order; then for each maximal run of consecutive positive points $x_j, x_{j+1}, ...x_{j+k}$, defines the interval $[x_j, x_{j+k}]$; and finally, outputs as h_S the union of these intervals. Clearly, $h_S \in \mathcal{H}_s$, which gives $\mathcal{H}^{\mathcal{A}}_{s,m} = \mathcal{H}_s$ with $\text{VCdim}(\mathcal{H}^{\mathcal{A}}_{s,m}) = \text{VCdim}(\mathcal{H}_s) = 2s$, and moreover, $\text{er}_S[h_S] = 0$. The algorithm runs in $O(m \ln m)$ time. Therefore, by Theorem 2.3, $\mathcal{H}$ is *efficiently learnable w.r.t. the target complexity function* size.

3 Necessary Conditions for Learnability and Lower Bounds on Sample Complexity

The following establishes a lower bound on the sample complexity of learning a binary-valued function class; see for example [2] or [1] for a proof:

Theorem 3.1 (Ehrenfreucht et al, 1989; Anthony and Bartlett, 1999)**.** Let $\mathcal{H} \subseteq \{\pm 1\}^{\mathcal{X}}$ with $\text{VCdim}(\mathcal{H}) = d$. Then:

(1) (General/agnostic setting) Any algorithm $\mathcal{A}$ that (ϵ, δ)-learns $\mathcal{H}$ for all $\epsilon, \delta \in (0, 1]$ and all distributions D on $\mathcal{X} \times \{\pm 1\}$ has sample complexity

$$m_{\mathcal{A}}(\epsilon, \delta) \geq \max\left(\frac{d}{320\epsilon^2}, \frac{1-\epsilon^2}{\epsilon^2} \ln\left(\frac{1}{8\delta(1-2\delta)}\right)\right) \tag{6}$$

for small enough ϵ, δ.

(2) (Target function setting) Any algorithm $\mathcal{A}$ that (ϵ, δ)-learns $\mathcal{H}$ in the target function setting for all $\epsilon, \delta \in (0, 1]$ and all distributions μ on $\mathcal{X}$ and target functions $t \in \mathcal{H}$ has sample complexity

$$m_{\mathcal{A}}(\epsilon, \delta) \geq \max\left(\frac{d-1}{32\epsilon}, \frac{1}{2\epsilon} \ln\left(\frac{1}{\delta}\right)\right) \tag{7}$$

for small enough ϵ, δ.

4 Characterizations of Learnability

The following characterization of learnability of a binary-valued function class follows directly from Theorems 1 and 4:

Corollary 4.1. A binary-valued function class $\mathcal{H} \subseteq \{\pm 1\}^{\mathcal{X}}$ is learnable (in the general or target function setting) iff $\text{VCdim}(\mathcal{H})$ is finite.

Remark. An analogous result holds for real-valued function classes: a bounded, real-valued function class $\mathcal{F} \subseteq [-1, 1]^{\mathcal{X}}$ is learnable (w.r.t. squared loss) iff $\text{fat}_{\mathcal{F}}(\epsilon)$ is finite for all $\epsilon > 0$. See for example [3] or [1] for more details.

The following provides a characterization of *efficient* learnability of a binary-valued function class:

Theorem 4.2. Let $\mathcal{H} \subseteq \{\pm 1\}^{\mathcal{X}}$. Then:

(1) (General/agnostic setting) $\mathcal{H}$ is efficiently learnable iff VCdim($\mathcal{H}$) is finite and there exists a randomized algorithm $\mathcal{A}$ that given $S \in (\mathcal{X} \times \{\pm 1\})^m$, halts in poly($m$) time and with probability $\geq \frac{1}{2}$, returns $h_S \in \arg\min_{h \in \mathcal{H}}(\mathrm{er}_S[h])$.

(2) (Target function setting) $\mathcal{H}$ is efficiently learnable in the target function setting iff VCdim($\mathcal{H}$) is finite and there exists a randomized algorithm $\mathcal{A}$ that given $S \in (\mathcal{X} \times t)^m$ for any $t \in \mathcal{H}$, halts in poly(m) time and with probability $\geq \frac{1}{2}$, returns $h_S \in \mathcal{H}$ with $\mathrm{er}_S[h_S] = 0$.

Proof. (sketch) We'll consider the general setting (1); the argument for the target function setting (2) is similar. For the 'if' direction, suppose VCdim($\mathcal{H}$) is finite and there exists a randomized algorithm $\mathcal{A}$ as specified. Then construct a randomized algorithm $\mathcal{B}$ which, given a training sample $S \in (\mathcal{X} \times t)^m$, runs $\mathcal{A}$ on this sample r times (independent runs) and returns the function with the lowest training error. Then clearly $\mathcal{B}$ runs in poly(mr) time and w.p. $\geq (\frac{1}{2})^r$, returns a function $h_S \in \arg\min_{h \in \mathcal{H}}(\mathrm{er}_S[h])$. Given learning parameters ϵ, δ, it can be verified that taking $r = \lceil \log_2 \frac{2}{\delta} \rceil$ and selecting the sample size m in Theorem 2.1 as a function of ϵ and $\frac{\delta}{2}$ yields the desired result. For the 'only if' direction, suppose $\mathcal{H}$ is efficiently learnable. We know from Corollary 4.1 that VCdim($\mathcal{H}$) is finite. To construct a randomized algorithm $\mathcal{A}$ with the desired properties, we can simply take $\epsilon = \frac{1}{2m}$ and $\delta = \frac{1}{2}$, and take D to be the uniform distribution on the examples in the training sample S; then constructing a random sample S' of the same size m but with examples drawn iid from S and running the learning algorithm for $\mathcal{H}$ with the chosen parameters ϵ, δ on S' yields the desired result. For more details see for example [1]. □

The following result provides an analogous characterization of efficient learnability w.r.t. domain dimension:

Theorem 4.3. Let $\mathcal{H} = \cup_{n=1}^{\infty} \mathcal{H}_n$, where $\mathcal{H}_n \subseteq \{\pm 1\}^{\mathcal{X}_n}$ and instances in $\mathcal{X}_n$ can be encoded using a representation of size polynomial in n (typically, $\mathcal{X}_n \subseteq \mathbb{R}^n$). Then:

(1) (General/agnostic setting) The dimension-graded class $\mathcal{H} = \cup_{n=1}^{\infty} \mathcal{H}_n$ is efficiently learnable iff VCdim($\mathcal{H}_n$) = poly(n) and there exists a randomized algorithm $\mathcal{A}$ that given $S \in (\mathcal{X}_n \times \{\pm 1\})^m$, halts in poly($m, n$) time and w.p. $\geq \frac{1}{2}$, returns $h_S \in \arg\min_{h \in \mathcal{H}_n}(\mathrm{er}_S[h])$.

(2) (Target function setting) The dimension-graded class $\mathcal{H} = \cup_{n=1}^{\infty} \mathcal{H}_n$ is efficiently learnable in the target function setting iff VCdim($\mathcal{H}_n$) = poly(n) and there exists a randomized algorithm $\mathcal{A}$ that given $S \in (\mathcal{X}_n \times t)^m$ for any $t \in \mathcal{H}_n$, halts in poly(m, n) time and w.p. $\geq \frac{1}{2}$, returns $h_S \in \mathcal{H}_n$ with $\mathrm{er}_S[h_S] = 0$.

Example 7 (Linear classifiers in $\mathbb{R}^n$, continued from Example 5)**.** Since ERM in this case is NP-hard, the (dimension-graded) class of linear classifiers over $\mathbb{R}^n$ is *not efficiently learnable in the general setting (unless* RP = NP*).* As noted in Example 5, linear classifiers are efficiently learnable in the target function setting.

Example 8 (Boolean linear classifiers/Boolean threshold functions over $\{0,1\}^n$)**.** Let $\mathcal{X}_n = \{0,1\}^n$, and let

$$\mathcal{H}_n = \{h : \mathcal{X}_n \to \{\pm 1\} \mid h(\mathbf{x}) = \mathrm{sign}(\mathbf{w} \cdot \mathbf{x} - b) \text{ for some } \mathbf{w} \in \{0,1\}^n, b \in \mathbb{R}\}.$$

As shown by Pitt and Valiant, given a sample $S \in (\mathcal{X}_n \times t)^m$ for some $t \in \mathcal{H}_n$, finding a function $h_S \in \mathcal{H}_n$ with $\mathrm{er}_S[h_S] = 0$ is an NP-hard problem [4]. Therefore, by Theorem 4.3, $\mathcal{H} = \cup_{n=1}^{\infty} \mathcal{H}_n$ is *not (properly) efficiently learnable in the target function setting (unless* RP = NP*).* However, we also have

$$\mathcal{H}_n \subseteq \mathcal{H}'_n = \{h : \mathcal{X}_n \to \{\pm 1\} \mid h(\mathbf{x}) = \mathrm{sign}(\mathbf{w} \cdot \mathbf{x} - b) \text{ for some } \mathbf{w} \in \mathbb{R}^n, b \in \mathbb{R}\}.$$

For $\mathcal{H}'_n$, we know that there exists an efficient (poly(m, n) time) algorithm that, given $S \in (\mathcal{X}_n \times t)^m$ for any $t \in \mathcal{H}_n$, finds $h_S \in \mathcal{H}'_n$ with $\mathrm{er}_S[h_S] = 0$ (and moreover, VCdim($\mathcal{H}'_n$) $\leq n + 1$). Therefore, $\mathcal{H}$ is *efficiently learnable by* $\cup_{n=1}^{\infty} \mathcal{H}'_n$ *in the target function setting.*

Example 9 (k-term DNF/k-clause CNF over $\{0,1\}^n$)**.** Let $\mathcal{X}_n = \{0,1\}^n$, and let $k \geq 2$ be a constant. Let $\mathcal{H}_n$ be the set binary-valued of functions on $\mathcal{X}_n$ that can be represented as a k-term DNF formula (i.e. a DNF formula with at most k terms) over $\mathcal{X}_n$. Then as shown by Pitt and Valiant, given a sample $S \in (\mathcal{X}_n \times t)^m$ for some $t \in \mathcal{H}_n$, finding a function $h_S \in \mathcal{H}_n$ with $\mathrm{er}_S[h_S] = 0$ is an NP-hard problem [4]. Therefore, by Theorem 4.3, $\mathcal{H} = \cup_{n=1}^{\infty} \mathcal{H}_n$ is *not (properly) efficiently learnable in the target function setting (unless* RP = NP*).* However, we also have $\mathcal{H}_n \subseteq \mathcal{H}'_n$ where $\mathcal{H}'_n$ is the set of functions that can be represented as a k-CNF formula. For $\mathcal{H}'_n$, we know that there exists an efficient (poly(m, n) time) algorithm that, given $S \in (\mathcal{X}_n \times t)^m$ for any $t \in \mathcal{H}_n$, finds $h_S \in \mathcal{H}'_n$ with $\mathrm{er}_S[h_S] = 0$ (and moreover, VCdim($\mathcal{H}'_n$) = poly(n)). Therefore, $\mathcal{H}$ is *efficiently learnable by* $\cup_{n=1}^{\infty} \mathcal{H}'_n$ *in the target function setting.*

5 Next Lecture

Over the next two lectures, we will consider learnability of decision trees over the Boolean hypercube $\{0,1\}^n$. Specifically, in the next lecture, we will define a complexity measure termed the 'rank' of a decision tree, and will see that (a) the VC-dimension of the set of all binary-valued functions on $\{0,1\}^n$ that can be represented as a decision tree of rank at most r (for $r < n$) is $O(n^r)$; and (b) there is an algorithm that given any sample $S \in (\{0,1\}^n \times \{\pm 1\})^m$ labeled by a decision tree of rank at most r, returns in time $O(m(n+1)^{2r})$ a decision tree of rank at most r that has zero training error on S. Thus for any fixed r, this will give that decision trees of rank at most r are efficiently (properly) learnable in the target function setting. We will also see that a decision tree of size (number of nodes) s has rank $O(\ln s)$, so that this gives a quasi-polynomial time algorithm for learning decision trees of size at most s (time complexity $O(n^{O(\ln s)}, \frac{1}{\epsilon}, \ln(\frac{1}{\delta}))$). In the lecture after that, we will see that if we are interested in learnability only with respect to the uniform distribution over $\{0,1\}^n$, do not require proper learnability (i.e. do not require the learned function to be represented as a size-s decision tree), and the algorithm is allowed to make membership queries (i.e. is allowed to request the value of the target function on specific points in $\{0,1\}^n$), then using a different technique, namely that of Fourier analysis on the Boolean hypercube, it is possible to obtain a polynomial time algorithm for learning size-s decision trees (time complexity $O(\text{poly}(n,s), \frac{1}{\epsilon}, \ln(\frac{1}{\delta}))$).

References

[1] Martin Anthony and Peter L. Bartlett. *Learning in Neural Networks: Theoretical Foundations*. Cambridge University Press, 1999.

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

[3] Michael J. Kearns, Robert E. Schapire, and Linda Sellie. Toward efficient agnostic learning. *Machine Learning*, 17(2-3):115–141, 1994.

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