Leslie G. Valiant
TFs: Aayush Karan and Kevin Cong

CS 2280: Computational Learning Theory
Homework 3 Solutions
Due: Mar. 28, 11:59pm

Policy reminders You are strongly encouraged to type your solutions using LaTeX. You may discuss problems with your classmates, but not merely copy each others solutions. You must write all solutions by yourself, list your collaborators on your problem sets and also appropriately cite any resources outside of the class materials that you have used. You are not allowed to look up solutions to the problems. Please do not use LLMs or LLM-assisted tools for finding solutions to the problems.

Problem 1. (10pt) **Saturating Sauer-Shelah.** Show that for any d there is a concept class $\mathcal{C}$ of VC dimension d such that for any m there exists a set S of m points such that $|\Pi_{\mathcal{C}}(S)| = \Phi_d(m)$.

Sauer–Shelah Lemma. The Sauer–Shelah Lemma states that for a concept class $\mathcal{C}$ of VC dimension d, and any finite set S of m points,

$$|\Pi_{\mathcal{C}}(S)| \;\le\; \Phi_d(m) \;=\; \sum_{i=0}^{d} \binom{m}{i}.$$

We want to exhibit, for each fixed d, a concept class whose number of dichotomies exactly matches $\Phi_d(m)$ for every subset S of size m. In other words, we seek a class $\mathcal{C}$ of VC dimension d so that

$$\big|\Pi_{\mathcal{C}}(S)\big| \;=\; \Phi_d(m) \quad \text{for all } m \text{ and all } S \text{ with } |S| = m.$$

Construction of the saturating class. Let X be an infinite domain (or at least sufficiently large). Define

$$\mathcal{C} \;=\; \{\, c_T \;:\; T \subseteq X,\; |T| \le d \,\},$$

where each "concept" c_T is the indicator function of the finite set T, that is,

$$c_T(x) \;=\; \begin{cases} 1, & x \in T, \\ 0, & x \notin T. \end{cases}$$

In other words, $\mathcal{C}$ consists of all subsets of X whose size is at most d, interpreted as characteristic functions.

Why $\mathcal{C}$ has VC dimension d. First observe that no set of size larger than d can be shattered, since each concept in $\mathcal{C}$ picks out at most d points. Hence

$$\text{VC-dim}(\mathcal{C}) \;\le\; d.$$

On the other hand, any set S^* of size d is shattered by $\mathcal{C}$: for any chosen subset $A \subseteq S^*$ (of any size up to d), we can take the concept c_A to pick out exactly A from S^*. This shows

$$\text{VC-dim}(\mathcal{C}) \;\ge\; d.$$

Thus $\text{VC-dim}(\mathcal{C}) = d$.

Why $|\Pi_{\mathcal{C}}(S)| = \Phi_d(|S|)$. Take any set $S \subseteq X$ of size m. Then

$$\Pi_{\mathcal{C}}(S) \;=\; \{\, c_T\!\restriction_S \mid\; T \in \mathcal{C}\} \;=\; \{\, T \cap S \;\mid\; |T| \le d\}.$$

But $T \cap S$ is any subset of S of size at most d. Hence

$$\bigl|\Pi_{\mathcal{C}}(S)\bigr| \;=\; \sum_{i=0}^{d} \binom{m}{i} \;=\; \Phi_d(m),$$

exactly saturating the Sauer–Shelah bound for all m. Therefore, for every set S of m points,

$$\bigl|\Pi_{\mathcal{C}}(S)\bigr| \;=\; \Phi_d(m),$$

as desired.

Conclusion. This family of "indicator-of-T" concepts thus has VC dimension d and achieves the maximum possible number of dichotomies $\Phi_d(m)$ on every m-element subset S, thereby saturating the bound stated in the problem.

Problem 2. (10pt) **Monotone Boolean functions are not PAC-learnable.** Let $x = x_1 \ldots x_n \in \{0,1\}^n$ and $y = y_1 \ldots y_n \in \{0,1\}^n$ be two n-bit strings. We say that $x \ge y$ if $x_i \ge y_i$ for all i. A boolean function $f : \{0,1\}^n \to \{0,1\}$ on variables $x_1, \ldots, x_n$ is said to be *monotone* if $x \ge y$ implies $f(x) \ge f(y)$.
Let M_n denote the class of all monotone Boolean functions over $\{0,1\}^n$. Prove that there is no PAC learning algorithm for M_n whose running time is a polynomial function of $n, \frac{1}{\epsilon}, \frac{1}{\delta}$ (without size(c)).

Problem Statement. Let M_n be the class of all monotone Boolean functions on n variables, that is, all $f : \{0,1\}^n \to \{0,1\}$ satisfying

$$x \ge y \quad\Longrightarrow\quad f(x) \;\ge\; f(y) \quad \text{for all}\, x, y \in \{0,1\}^n.$$

We want to show that *no* polynomial-time PAC learning algorithm can learn M_n *unless* it is allowed to depend on the size of the target concept. In other words, there is no algorithm that runs in time

$$\mathrm{poly}\bigl(n,\; 1/\varepsilon,\; 1/\delta\bigr)$$

that PAC learns *all* monotone functions of n variables (when the algorithm's runtime is not permitted to depend on size(c)).

Key idea: 3-Term DNF is a small subfamily of monotone Boolean functions.

- 3-Term DNF formulas are already hard to PAC learn, in the sense that learning 3-Term DNF by 3-Term DNF is NP-hard.
- Every 3-Term DNF formula is monotone because it has no negated variables.

Hence the class of 3-Term DNF embeds into M_n: every 3-Term DNF is in fact a monotone Boolean function on n variables.

Hardness argument. Suppose, for contradiction, that there were a PAC learner $\mathcal{A}$ that learns M_n in time poly$(n, 1/\varepsilon, 1/\delta)$ with no dependence on size(c). Then:

1. Given any target concept c that is specifically a 3-Term DNF (hence monotone), we can feed $\mathcal{A}$ those labeled examples.
2. Because 3-Term DNF $\subseteq M_n$, the hypothesized learner $\mathcal{A}$ would succeed (by assumption) in polynomial time in $n, 1/\varepsilon, 1/\delta$.

But from standard NP-hardness reductions, learning 3-Term DNF in polynomial time is known to be NP-hard unless RP = NP. Thus we get a contradiction.

Hence no such polynomial-time learner for the entire class M_n can exist, if the runtime is forbidden from depending on size(c). Concretely, the sheer complexity of even small monotone formulas (e.g. 3-term DNF) forces a computational hardness result.

Problem 3. (10pt) **VC-dimension of parity functions.** Define the class of parity functions $\mathcal{P}$ over $X = \{0,1\}^n$ as follows: Let $a \in \{0,1\}^n$, then $\chi_a(x) = 1$ if $a \cdot x$ is odd and $\chi_a(x) = 0$ otherwise, where $a \cdot x = \sum_{i=1}^{n} a_i x_i$. Prove that the VC-dimension of $\mathcal{P}$ is n.

Problem Statement. Let P be the class of all parity functions on n bits. For each $a \in \{0,1\}^n$, define the parity function

$$\chi_a(x) \;=\; \begin{cases} 1, & \text{if } a \cdot x \text{ is odd,} \\ 0, & \text{if } a \cdot x \text{ is even,} \end{cases} \qquad \text{where} \quad a \cdot x \;=\; \sum_{i=1}^{n} a_i\, x_i \pmod 2.$$

We claim that $\mathrm{VCdim}(P) = n$.

1. VC-dimension is at least n. To show any set S of n points in $\{0,1\}^n$ can be shattered, we choose S to be linearly independent over GF(2). (For instance, we can pick S to be the standard basis vectors $e_1, \ldots, e_n$ in $\{0,1\}^n$)

Given any labeling $b : S \to \{0,1\}$ (so each point $x \in S$ is assigned label $b(x)$), we must find a parity function χ_a such that $\chi_a(x) = b(x)$ for all $x \in S$. Over GF(2), specifying $\chi_a(x) = b(x)$ is requiring $a \cdot x \equiv b(x) \pmod 2$. Because S is linearly independent, there is exactly one solution $a \in \{0,1\}^n$ to these $|S| = n$ linear equations in GF(2).

Hence, for every possible labeling b, there is exactly one parity function χ_a that matches b on all of S. Therefore S is shattered by P, giving $\mathrm{VCdim}(P) \geq n$.

2. VC-dimension is at most n. Any set of $n+1$ distinct points in $\{0,1\}^n$ cannot be linearly independent in $\mathbb{F}_2^n$, because the rank is at most n. Equivalently, if we attempt to label an $(n+1)$-element set in certain conflicting ways, no single parity function χ_a can match that labeling. Hence no subset of size $n+1$ can be shattered.

Conclusion. Combining the two parts yields

$$\mathrm{VCdim}(P) \;=\; n.$$

Thus the class of parity functions on n bits has exactly VC-dimension n.

Problem 4. (10pt) **Compositional VC Dimension.** Let $\mathcal{C}$ be a concept class over some domain X and $\mathcal{F}_T$ be a concept class over $\{0,1\}^T$. We define a class of functions $\mathcal{F}_T(\mathcal{C})$ over X as follows.

$$\mathcal{F}_T(\mathcal{C}) = \{g(c_1(x), c_2(x), ..., c_T(x)) | g \in \mathcal{F}_T \text{ and } c_1, ..., c_T \in C\}.$$

Prove that VC-dim$(\mathcal{F}_T(\mathcal{C})) = O(\ell \log \ell)$ where $\ell =$ VC-dim$(\mathcal{F}_T) + T\cdot$VC-dim$(\mathcal{C})$.

Problem Statement. We have two concept classes:

- A class C of Boolean concepts over X, with $\text{VCdim}(C) = d_C$.
- A class F_T of Boolean concepts over $\{0,1\}^T$, with $\text{VCdim}(F_T) = d_F$.

We form a new concept class

$$F_T(C) \;=\; \Big\{ g\big(c_1(x), \ldots, c_T(x)\big) \;\mid\; g \in F_T,\;\; c_1, \ldots, c_T \in C \Big\}.$$

We need to show

$$\text{VCdim}\big(F_T(C)\big) \;=\; O\big(\ell \log \ell\big), \quad \text{where} \quad \ell \;=\; d_F \;+\; T\, d_C.$$

Strategy. The idea is a two-stage analysis: 1. First, each $x \in X$ is mapped to the T-bit tuple $\big(c_1(x), \ldots, c_T(x)\big)$. Because each c_i comes from a class of VC dimension d_C, we can see that only so many labelings on a finite sample can arise before we exceed $T\, d_C$ in dimension constraints. 2. Then we apply $g \in F_T$, whose own VC dimension is d_F, on the T-bit outputs.

We combine these via a known composition bound. Concretely, if you have one concept class $\mathcal{H}_1$ of dimension d_1 and another class $\mathcal{H}_2$ of dimension d_2, the composed class often has dimension $O\big((d_1 + d_2) \log(d_1 + d_2)\big)$. In our situation:

$$d_1 \;=\; d_F, \quad d_2 \;=\; T\, d_C, \quad \ell = d_1 + d_2.$$

Hence we obtain

$$\text{VCdim}\big(F_T(C)\big) \;=\; O\big(\ell \,\log \ell\big).$$

Sketch of the combinatorial argument. If $F_T(C)$ shattered a large set of points in X, that would imply that for each labeling of those points, we can pick some T concepts $c_1, \ldots, c_T \in C$ and some aggregator $g \in F_T$ that realizes exactly that labeling. But:

1. By the $\text{VCdim}(C) = d_C$ bound, once the sample set is too big, we cannot freely choose so many distinct ways to produce T different bit-tuples $\big(c_1(x), \ldots, c_T(x)\big)$ on that sample without hitting a dimension limit of about $T\, d_C$.
2. Even if we did manage all those bit-tuples, to label them using $g \in F_T$ with $\text{VCdim}(F_T) = d_F$ we get another dimension constraint.

A known "compositional bounding" approach merges these constraints into $O\big((d_F + T\, d_C) \log(d_F + T\, d_C)\big)$.

Conclusion. Hence we conclude

$$\mathrm{VCdim}\big(F_T(C)\big) \;\leq\; O\big((d_F + T\, d_C)\, \log(d_F + T\, d_C)\big).$$

Equivalently, if we define $\ell = d_F + T\, d_C$, we get

$$\mathrm{VCdim}\big(F_T(C)\big) \;=\; O\big(\ell \, \log \ell\big),$$

as required.

Problem 5. (15pt) **Occam as a weak learning algorithm.** Let $\mathcal{C}$ be any concept class. Show that if there exists an (α, β)-Occam algorithm for $\mathcal{C}$, then there exists an efficient randomized Occam algorithm that given sample S of size m for $c \in C$, with probability at least $1 - \delta$, outputs a hypothesis h consistent with S such that $\mathrm{size}(h) \leq p(n, size(c), \log m)$ for some polynomial p.

Hint. You can assume here that the description of each real number that occurs in an execution of the Adaboost algorithm takes $O(1)$ space.

Problem 6. (15pt) **Adaboost on weak learning algorithm.** Let $\mathcal{C}$ be a concept class and `WeakLearn` be an algorithm that weakly PAC learns $\mathcal{C}$ and generates hypotheses that have error of at most $1/2 - \gamma$ for some positive γ (assume for simplicity that `WeakLearn` always succeeds). Let x be any point in the sample S of size $N \geq 2$.

1. Show that in the Adaboost algorithm, the error of hypothesis h_t on distribution D_{t+1} is exactly 1/2.
2. What is the maximum probability that the Adaboost algorithm can assign to point x in any of the boosting stages?
3. Assuming that `WeakLearn` fails, for as long as it possibly can, to return the correct label for x, what is the maximum number of stages that it will take the Adaboost algorithm to force `WeakLearn` to return a hypothesis which is correct on x (give the best upper bound you can). *You can assume that initially every point has the same probability.*