CPSC 532D — 7. LOWER BOUNDS / NO FREE LUNCH

Danica J. Sutherland
University of British Columbia, Vancouver

Fall 2023

So far we've done upper bounds: in this setting, we know we can learn at least this well. But if we only know upper bounds, we never really know how tight they are, and so we can never really know if one algorithm is better than another.

One way to approach this is with asymptotic results, as described e.g. by [Bach23, Section 4.6] who summarizes and translates results from the classic textbook of van der Vaart [vdV98]. For instance, if $\mathcal{H} = \{h_w : w \in \mathcal{W}\}$ for some open $\mathcal{W} \subseteq \mathbb{R}^D$, the loss is sufficiently "nice" as a function of w, and there's a minimizer $h^* = h_{w^*}$, then with some extra "niceness" assumptions we have for the ERM that

$$\mathop{\mathbb{E}}_{S\sim\mathcal{D}^m} L_\mathcal{D}(\hat{h}_S) - L_\mathcal{D}(h^*) = \Theta\left(\frac{1}{m}\operatorname{Tr}\left(\left[\nabla_w^2 L_\mathcal{D}(h^*)\right]^{-1} \mathop{\mathbb{E}}_{z\sim\mathcal{D}}\left[(\nabla_w \ell(h_w, z))(\nabla_w \ell(h_w, z)^\mathsf{T}|_{w=w^*}\right]\right)\right).$$

This actually gives a fast $1/m$ rate, and along the way it actually also tells us that $w - w^*$ is asymptotically Gaussian, and some other nice things. If we can evaluate the stuff inside the trace, we could also then explicitly say "this $\mathcal{H}$ converges faster than that one," or compare to an asymptotic rate for some different algorithm. But: the "niceness" assumptions don't always hold, the expressions aren't always easy to analyze, and they're purely asymptotic results, so we don't know whether they're a good approximation after $m = 20$ or only after $m = 100,000,000,000$.

Instead, let's use a different route to lower bounds, specifically for binary classifiers.

1 NO FREE LUNCH FOR HIGH-VC CLASSES

THEOREM 1. *Let $\mathcal{H}$ be a hypothesis set of binary classifiers over $\mathcal{X}$. Let $m \le \operatorname{VCdim}(\mathcal{H})/2$. Then*

$$\inf_{\mathcal{A}} \sup_{\mathcal{D} \text{ realizable by } \mathcal{H}} \Pr_{S\sim\mathcal{D}^m}\left(L_\mathcal{D}(\mathcal{A}(S)) \ge \frac{1}{8}\right) \ge \frac{1}{7},$$

where the infimum over $\mathcal{A}$ is over all learning algorithms which return hypotheses in $\mathcal{H}$.

This theorem is similar to Theorem 5.1 of [SSBD], but incorporating the idea of VC dimension (which they haven't introduced yet at that point).

Before we prove this, let's unpack the quantifiers a bit. For any m and any learning algorithm $\mathcal{A}$, there is some realizable distribution $\mathcal{D}$ such that $\mathcal{A}$ has at least constant probability of failing with m samples, i.e. getting at least $1/8$ error. Note that this distribution *depends on m*, and also on $\mathcal{A}$. This immediately implies the following:

COROLLARY 2. *Any $\mathcal{H}$ with $\operatorname{VCdim}(\mathcal{H}) = \infty$ is not PAC learnable.*

Proof of Theorem 1. We're first going to pick a shatterable set of size $2m$, $\tilde{\mathcal{X}} = \{\tilde{x}_1, \ldots, \tilde{x}_{2m}\} \subseteq \mathcal{X}$; at least one such set must exist, since $2m \le \operatorname{VCdim}(\mathcal{H})$. Then we'll pick the marginal distribution of x, $\mathcal{D}_x$, to be a discrete uniform distribution on $\tilde{\mathcal{X}}$.

Since we're being totally generic with respect to $\mathcal{A}$, it's going to be hard to say which $y \mid x$ labeling rule in particular is going to be hard for $\mathcal{A}$ to learn. So, as a proof

For more, visit https://cs.ubc.ca/~dsuth/532D/23w1/.

technique, we're going to start with a *random* f, and then settle on a particular one later. Specifically, for each vector of possible labels $y \in \{0,1\}^m$, choose some particular $f \in \mathcal{H}$ such that $f(x_j) = y_j$; there must be at least one, since $\mathcal{H}$ shatters $\tilde{\mathcal{X}}$. Let $\mathcal{F}$ be the set of these functions (of size exactly 2^m), and choose $f \sim \operatorname{Unif}(\mathcal{F})$. For any f, let the distribution $\mathcal{D}_{(f)}$ denote the distribution that you get by sampling $x \sim \mathcal{D}_x$, $y \mid x = f(x)$.

Now, for any sample of inputs $\mathrm{S}_x = (x_1, \dots, x_m)$, we can implicitly construct a sample of pairs $\mathrm{S} = ((x_1, f(x_1)), \dots, (x_m, f(x_m)))$; call the result of the algorithm $\hat{h}_\mathrm{S} = \mathcal{A}(\mathrm{S})$. Its expected loss is

$$\mathop{\mathbb{E}}_{f \sim \operatorname{Unif}(\tilde{\mathcal{X}} \to \mathcal{Y})} \mathop{\mathbb{E}}_{\mathrm{S} \sim \mathcal{D}_{(f)}^m} \mathrm{L}_{\mathcal{D}_{(f)}}(\hat{h}_\mathrm{S}) = \mathop{\mathbb{E}}_{f, \mathrm{S}_\mathrm{X}} \mathop{\mathbb{E}}_{x \sim \mathcal{D}_x} \mathbb{1}(\hat{h}_\mathrm{S}(x) \neq f(x)).$$

Using the law of total expectation, let's break this expectation up based on whether the test x is in the training data S or not:

$$\begin{aligned} \mathop{\mathbb{E}}_{f, \mathrm{S}_x, x} \mathbb{1}(\hat{h}_\mathrm{S}(x) \neq f(x)) = \mathop{\mathbb{E}}_{f, \mathrm{S}_x} \Big[&\Pr(x \notin \mathrm{S}_x) \mathop{\mathbb{E}}_{x \sim \mathcal{D}_x} [\mathbb{1}(\hat{h}_\mathrm{S}(x) \neq f(x)) \mid x \notin \mathrm{S}_x] \\ &+ \Pr(x \in \mathrm{S}_x) \mathop{\mathbb{E}}_{x \sim \mathcal{D}_x} [\mathbb{1}(\hat{h}_\mathrm{S}(x) \neq f(x)) \mid x \in \mathrm{S}_x] \Big]. \end{aligned}$$

For the second term, we're not going to worry about what the algorithm does on the data it's actually seen: we'll just bound this as being at least zero.

For the first term, we know since $\mathcal{D}_x$ is uniform and $|\mathrm{S}_x| \le m$ that

$$\Pr(x \notin \mathrm{S}_x) = \frac{\left|\tilde{\mathcal{X}} \setminus \mathrm{S}_x\right|}{\left|\tilde{\mathcal{X}}\right|} \ge \frac{m}{2m} = \frac{1}{2}.$$

Also, since our labels $f(\tilde{x}_j)$ are uniformly random and totally independent of one another, and S is statistically independent of those labels for points $\tilde{x} \notin \mathrm{S}$, whether $\hat{h}_\mathrm{S}$ agrees with f is just a pure coin flip: $\mathbb{E}_x[\mathbb{1}(\hat{h}_\mathrm{S}(x) \neq f(x)) \mid x \notin \mathrm{S}_x] = \frac{1}{2}$.

Combining, we know that

$$\mathop{\mathbb{E}}_{f \sim \operatorname{Unif}(\mathcal{X} \to \mathcal{Y})} \mathop{\mathbb{E}}_{\mathrm{S} \sim \mathcal{D}_{(f)}^m} \mathrm{L}_{\mathcal{D}_{(f)}}(\hat{h}_\mathrm{S}) \ge \tfrac{1}{4}.$$

But, if the *average* over functions f of the expected loss $\mathbb{E}_{\mathrm{S}_x \sim \mathcal{D}_x^n} \mathrm{L}_{\mathcal{D}_x, f}(\hat{h}_\mathrm{S})$ is at least $\frac{1}{4}$, then there must be at least one *particular* function f such that the expected loss is at least $\frac{1}{4}$! Pick one and call it g; this will be the labeling function claimed by the theorem.

This proof technique is known as the probabilistic method, *and often attributed to Paul Erdős.*

We've shown the average loss is large, but we want to show that the loss has high probability of being large. Now, $\mathrm{L}_{\mathcal{D}_{(g)}}(\hat{h}_\mathrm{S})$ is a random variable bounded in $[0,1]$, and we already know one way to bound those variables in terms of their means: Markov's inequality. But, unfortunately, Markov's inequality bounds the probability of things being *big*, and we want to bound the probability of this being *small*. So we'll need to switch it around, which is sometimes called "reverse Markov":

$$\Pr(\mathrm{L}_{\mathcal{D}_{(g)}}(\hat{h}_\mathrm{S}) \le \tfrac{1}{8}) = \Pr\left(1 - \mathrm{L}_{\mathcal{D}_{(g)}} \ge 1 - \frac{1}{8}\right) \le \frac{1 - \mathbb{E}\, \mathrm{L}_{\mathcal{D}_{(g)}}(\hat{h}_\mathrm{S})}{\frac{7}{8}} \le \left(1 - \frac{1}{4}\right)\frac{8}{7} = \frac{6}{7}.$$

Thus, for the realizable $\mathcal{D}_{(g)}$ we picked above,

$$\Pr_{S\sim\mathcal{D}_{(g)}^m}\left(L_{\mathcal{D}_{(g)}}(\hat{h}_S) > \tfrac{1}{8}\right) \geq \frac{1}{7}. \qquad \square$$

1.1 *Interpretation*

Theorem 1 is sometimes called a "no free lunch" theorem, in that there is no algorithm that *always* works (in the sense of PAC learning): every algorithm fails on at least one distribution.

In fact, basically this same proof strategy implies [Wol96] that, if you only care about the "off-sample" error (the average error on $(x, y) \mid x \notin S_x$), there are just as many possible distributions where your predictor is right as where it's wrong, regardless of your learning algorithm. If you don't assume *anything* about the world, all algorithms perform the same on average.

This is in some ways a deep philosophical problem, called the problem of induction and generally credited to David Hume. The fact that the sun rose every day so far doesn't, from "pure first principles," imply anything about whether it will rise tomorrow: we just decide to prefer "simple" explanations, i.e. we choose some $\mathcal{H}$ that we like. But that doesn't really answer which $\mathcal{H}$ would be good.

Actually, VC or Rademacher theory can't answer that problem either: it's preferable to choose a $\mathcal{H}$ with small complexity, but since $\operatorname{Rad}((\mathcal{H} + \{f\})|_S) = \operatorname{Rad}(\mathcal{H}|_S)$, and $\operatorname{VCdim}(\mathcal{H}) = \operatorname{VCdim}(\{x \mapsto h(x)f(x) : h \in \mathcal{H}\})$ for ±1-valued h and f, we haven't actually seen any objective notion of a "simple hypothesis": only ways to say that *sets* of hypotheses are all similar enough to one another.

Sometimes people get a little mystical about no free lunch theorems, though – e.g. `https://no-free-lunch.org` says that this result "calls the whole of science into question." But the world is *not* uniformly random; we know from experience that some kinds of $\mathcal{H}$ tend to work better than others. So, although there is *some* distribution that every algorithm fails on, it's not the case in the world we live in that all algorithms are the same as each other. (And, interestingly, there *are* (impractical) learning algorithms that are always at least as good as any other algorithm, up to (huge) constants: check out `https://free-lunch.org` [Nak21].)

2 LOWER BOUNDS

We can also use Theorem 1 to prove a quantitative lower bound on learning with *any* m and d:

THEOREM 3. *Let $\mathcal{H}$ be a set of binary classifiers over $\mathcal{X}$. For any $m \geq 1$,*

$$\inf_{\mathcal{A}} \sup_{\mathcal{D} \text{ realizable by } \mathcal{H}} \Pr_{S\sim\mathcal{D}^m}\left(L_{\mathcal{D}}(\mathcal{A}(S)) > \frac{\operatorname{VCdim}(\mathcal{H}) - 1}{32m}\right) \geq \frac{1}{100}$$

where $L_{\mathcal{D}}$ uses zero-one loss, and the infimum over $\mathcal{A}$ is over all learning algorithms returning hypotheses in $\mathcal{H}$.

This statement follows [MRT, Theorem 3.20], which merges this result and Theorem 1 in a way I find really *hard to follow, and can't handle the high-VC case. [SSBD, Theorem 6.8] states a similar result but leaves this part as an exercise.*

Proof. Let $d = \operatorname{VCdim}(\mathcal{H})$; note that if $d = 1$, the result holds trivially, so assume $d \geq 2$. Also, if $m \leq d/2$, then $\frac{d-1}{32m} \geq \frac{1}{16}\frac{d-1}{d} \geq \frac{1}{32}$; Theorem 1 says that there's at least $\frac{1}{7}$ probability of the error being at least $\frac{1}{8}$, which necessarily implies there's at least

$\frac{1}{7}$ probability of the error being over $\frac{1}{32}$, and so there's at least $\frac{1}{100}$ probability of the error being over $\frac{1}{32}$. Thus, assume that $m > d/2$.

Choose a set $\tilde{\mathcal{X}} = \{\tilde{x}_1, \dots, \tilde{x}_d\}$ of size $d = \mathrm{VCdim}(\mathcal{H})$ which can be shattered by $\mathcal{H}$. We're going to choose a distribution that puts most of its probability mass on $\tilde{x}_1$, in such a way that we're likely to see less than half of the *other* points from the distribution. Specifically, for an $\varepsilon > 0$ to choose later,

$$\Pr_{x \sim \mathcal{D}_x}(x = \tilde{x}_1) = 1 - \varepsilon, \qquad \text{for all } i > 1,\ \Pr_{x \sim \mathcal{D}_x}(x = \tilde{x}_i) = \frac{\varepsilon}{d-1}.$$

Now, let $\tilde{\mathcal{D}}$ be the distribution over $\{\tilde{x}_2, \dots, \tilde{x}_d\}$ selected by Theorem 1 with $m = (d-1)/2$, and let $f \in \mathcal{H}$ be the labeling function chosen in $\tilde{\mathcal{D}}$. Our distribution will be found by sampling $x \sim \mathcal{D}_x$ and then letting $y \mid x = f(x)$.

Now, we're going to prove that it's fairly likely that samples from $\mathcal{D}_x$ contain at most $(d-1)/2$ of the non-$\tilde{x}_1$ points. How many points we *don't* see is a little annoying to characterize exactly, but we can get a bound based on

$$Q = \sum_{i=1}^{m} \mathbb{1}(x_i \neq \tilde{x}_1);$$

if we repeat any of the non-$\tilde{x}_1$ points, Q will double-count them, but it's a valid upper bound on the number of non-$\tilde{x}_1$ points we see. Notice that $\Pr(x_i \neq \tilde{x}_1) = \varepsilon$, and each of the indicators is iid Bernoulli(ε), so $Q \sim \mathrm{Binomial}(m, \varepsilon)$.

A standard tail bound for binomial variables, Proposition 4 with $\gamma = 1$, shows that

$$\Pr(Q \geq 2m\varepsilon) \leq \exp\left(-\frac{1}{3} m\varepsilon\right).$$

To use this result, we want $2m\varepsilon = \frac{1}{2}(d-1)$; so, pick $\varepsilon = (d-1)/(4m)$. This is valid, since $m > d/2$ implies that $\varepsilon < \frac{1}{2}\frac{d-1}{d} < \frac{1}{2}$. Then we see less than half of the non-$\tilde{x}_1$ points with probability at least

$$1 - \exp\left(-\frac{m}{3} \cdot \frac{d-1}{4m}\right) = 1 - \exp\left(-\frac{d-1}{12}\right) \geq 1 - \exp\left(-\frac{1}{12}\right) > 0.07,$$

since $1 - \exp(-1/12) \approx 0.07995$.

So, with at least 7% probability, a sample of size m from $\mathcal{D}$ will contain at most $(d-1)/2$ of the non-$\tilde{x}_1$ points. Then, Theorem 1 tells us that with probability at least 1/7, $L_{\tilde{\mathcal{D}}}(\mathcal{A}(S)) \geq \frac{1}{8}$. If this happens, this implies that $L_{\mathcal{D}}(\mathcal{A}(S)) \geq \frac{1}{8}\varepsilon = \frac{d-1}{32m}$, since the total probability of the non-$\tilde{x}_1$ points is exactly ε. So, we have at least a $\frac{1}{7} \cdot 7\% = 1\%$ chance of seeing $\frac{d-1}{32m}$ error on $\mathcal{D}$, as desired. □

PROPOSITION 4. *If* $X \sim \mathrm{Binomial}(m, p)$*, then for any* $\gamma > 0$ *it holds that*

$$\Pr(X \geq (1+\gamma)mp) \leq \exp\left(-\frac{1}{3} mp\gamma^2\right).$$

This is an immediate consequence of the multiplicative Chernoff bound, which is e.g. Theorem D.4 of [MRT]. It's proved based on the same idea as Hoeffding/etc, just takes a little more work along those same lines.

Agnostic case You can get a bigger error if you don't require $\mathcal{D}$ to be realizable: Theorem 3.23 of [MRT] gives that for any m and $\mathcal{H}$,

$$\inf_{\mathcal{A}} \sup_{\mathcal{D}} \Pr\left(L_{\mathcal{D}}(\mathcal{A}(S)) - \inf_{h \in \mathcal{H}} L_{\mathcal{D}}(h) \geq \sqrt{\frac{d}{320m}} \right) \geq \frac{1}{64}. \tag{1}$$

Section 28.2 of [SSBD] is similar.

More generally These styles of theorems are sometimes called "minimax bounds," and algorithms are called "minimax-optimal" or simply "minimax" if they achieve the lower bound (usually only up to constants, though that's also sometimes called "rate-optimal"). In the VC notes we showed that ERM gets error $\widetilde{\mathcal{O}}_p(\sqrt{d/m})$, which combined with the agnostic result above shows that ERM is (up to log factors) rate-optimal for finite-VC classes. Although we haven't shown this (see Section 28.3 of [SSBD] or 6.5 of [Zhang23]), ERM for binary classifiers achieves $\widetilde{\mathcal{O}}_p(d/m)$ error in the realizable setting, so by Theorem 3 ERM is also (up to log factors) minimax rate-optimal for realizable distributions too.

Minimax rates are also available for various other problems, including things like linear regression, density estimation, and optimization. We won't talk a lot about lower bounds in this class, but they can be really nice to know whether your learning algorithm is "good" or not. (The problem, though, is they tend to be extremely "worst-case," and might not be too informative about problems you're likely to actually see – similar to no free lunch arguments.)

3 The "Fundamental Theorem of Statistical Learning"

We've now shown all the necessary parts for a pretty complete qualitative understanding of PAC learning for binary classifiers.

theorem 5 (Fundamental Theorem of Statistical Learning). *For $\mathcal{H}$ a class of functions $h : \mathcal{X} \to \{0,1\}$ and with the 0-1 loss, the following are equivalent:*

This name is only, as far as I know, used by [SSBD].

1. *Uniform convergence: for all $\varepsilon, \delta \in (0,1)$, we have that $\sup_{h \in \mathcal{H}} L_{\mathcal{D}}(h) - L_S(h) < \varepsilon$ with probability at least $1 - \delta$ as long as $m \geq m^{\mathrm{UC}}(\varepsilon, \delta) < \infty$.*
2. *Any ERM rule agnostically PAC-learns $\mathcal{H}$.*
3. *$\mathcal{H}$ is agnostically PAC learnable.*
4. *Any ERM rule PAC-learns $\mathcal{H}$.*
5. *$\mathcal{H}$ is PAC learnable.*
6. $\mathrm{VCdim}(\mathcal{H}) < \infty$.

[SSBD] use two-sided uniform convergence: in the setting of the theorem here, one-sided bounds imply two-sided ones, but (a) one-sided is what we really use, and (b) in more general settings the distinction can matter.

Proof. 1 implying 2 is our usual argument:

$$L_{\mathcal{D}}(\hat{h}_S) \leq L_S(\hat{h}_S) + \sup_{h \in \mathcal{H}} L_{\mathcal{D}}(h) - L_S(h) \leq L_S(h^*) + \varepsilon \leq L_{\mathcal{D}}(h^*) + [L_S(h^*) - L_{\mathcal{D}}(h^*)] + \varepsilon,$$

plus Hoeffding on $L_S(h^*) - L_{\mathcal{D}}(h^*)$.

2 implying 3, and 4 implying 5, are immediate.

2 implying 4, and 3 implying 5, is also straightforward from the definitions.

Corollary 2 shows that 5 implies 6.

6 implying 1 is implied by Theorem 10 of the VC notes, plus Theorem 8 from the Rademacher notes. □

Theorem 6.8 of [SSBD] gives a quantitative version, bounding the sample complexities in terms of the VC dimension, by collecting lower bounds like Theorem 3 and (1) and upper bounds like Theorem 10 of the VC notes and the realizable equivalent that we didn't prove.

References

[Bach23] Francis Bach. *Learning Theory from First Principles*. April 2023 draft.

[MRT] Mehryar Mohri, Afshin Rostamizadeh, and Ameet Talkwalkar. *Foundations of Machine Learning*. 2nd edition. MIT Press, 2018.

[Nak21] Preetum Nakkiran. *Turing-Universal Learners with Optimal Scaling Laws*. 2021. arXiv: 2111.05321.

[SSBD] Shai Shalev-Shwartz and Shai Ben-David. *Understanding Machine Learning: From Theory to Algorithms*. Cambridge University Press, 2014.

[vdV98] Aad W. van der Vaart. *Asymptotic Statistics*. Cambridge University Press, 1998.

[Wol96] David H. Wolpert. "The Lack of A Priori Distinctions Between Learning Algorithms." *Neural Computation* 8.7 (Oct. 1996), pages 1341–1390.

[Zhang23] Tong Zhang. *Mathematical Analysis of Machine Learning Algorithms*. 2023 pre-publication version.