CPSC 532D: Assignment 5 – due Saturday, 10 Dec 2022, **11:59pm**

As before: use LaTeX, either with the template I give or your own document if you prefer.

You can do this with a partner if you'd like (there's a "find a group" post on Piazza), but please **make sure you understand everything you're submitting** – don't just split an assignment in half. If you do parts of the assignment with a partner and parts separately, submit separate solutions, and say in each part you did together who you did it with.

If you look stuff up anywhere other than in **SSBD, MRT, Telgarsky, or Wainwright**, **cite your sources**: just say in the answer to that question where you looked. If you ask anyone else for help, **cite that too**. Please do not look at solution manuals / search for people proving the things we're trying to prove / etc. If you accidentally come across a solution while looking for something related, still write the argument up in your own words, link to wherever you found it, and be clear about what happened.

1 Validation sets and expectation bounds [20 points]

Stability and SGD analyses mostly bound only the expected risk; let's relate that a little more thoroughly to PAC learning now.

Let A be a proper learning algorithm (one returning hypotheses in $\mathcal{H}$) that guarantees: if $n \ge n_{\mathcal{H}}(\varepsilon)$, then for every distribution $\mathcal{D}$, it holds that

$$\mathop{\mathbb{E}}_{S \sim \mathcal{D}^n} L_{\mathcal{D}}(A(S)) \le \inf_{h \in \mathcal{H}} L_{\mathcal{D}}(h) + \varepsilon.$$

Here $L_{\mathcal{D}}(h) = \mathbb{E}_{z \sim \mathcal{D}}\, \ell(h, z)$ for a loss $\ell(h, z)$ bounded in $[0, 1]$.

In assignment 2 question 1b, we essentially showed (not quite in these words) via Markov's inequality that

$$\forall\, \varepsilon, \delta \in (0,1), \text{if } n \ge n_{\mathcal{H}}(\varepsilon\delta),\ \forall\, \mathcal{D}, \quad \Pr\left(L_{\mathcal{D}}(A(S)) \le \inf_{h \in \mathcal{H}} L_{\mathcal{D}}(h) + \varepsilon \right) \ge 1 - \delta. \tag{*}$$

(See the solutions for a2 if you want a reminder.)

(*) *implies PAC learning, but the dependence on δ is quite bad. For instance, the stability bound for regularized loss minimization (SSBD corollary 13.9) gives $n_{\mathcal{H}}(\varepsilon\delta) = \frac{8\rho^2 B^2}{\varepsilon^2 \delta^2}$, so going from a failure probability δ of 0.1 to 0.001 for a fixed ε requires 10,000 times as many samples, whereas for the bound we'll show below it's only three times as many samples.*

Here's a "meta-algorithm" that will PAC-learn with a better sample complexity:

- Divide the training data up into $k+1$ chunks, $S = S_1 \cup \cdots \cup S_k \cup V$, where each of the S_i have m data points and V has $n - mk$.
- Run A independently on each of the first k chunks, getting $\hat{h}_i = A(S_i)$.
- Let $\hat{\imath} = \arg\min_{i \in [k]} L_V(\hat{h}_i)$, and return the hypothesis $h_{\hat{\imath}}$. (That is, choose the "retry" that looks best on the validation set V.)

Give a way to choose the parameters m and k (that is, an expression for each variable depending only on on $n_{\mathcal{H}}$, ε, and δ) such that the procedure agnostically PAC-learns the problem, and give the final sample complexity. Your sample complexity should be $\mathcal{O}\left(\left[n_{\mathcal{H}}(a\varepsilon) + 1/\varepsilon^2\right] \log(1/\delta)\right)$ for some constant $a > 0$; please give a finite-sample expression, i.e. a function with explicit constants.

Answer: TODO

2 Logistic regression [30 points]

Let $\mathcal{X} = \{x \in \mathbb{R}^d : \|x\| \le R\}$. We'll learn a linear predictor based on logistic loss,

$$\ell^{\log}(w, (x, y)) = \lambda_y^{\log}(w^\mathsf{T} x) = \log(1 + \exp(-y w^\mathsf{T} x)).$$

Let S be a sample $((x_1, y_1), \dots, (x_n, y_n)) \in (\mathcal{X} \times \{-1, 1\})^n$, and let $X \in \mathbb{R}^{n \times d}$, $y \in \{-1, 1\}^n$ stack up the features and labels accordingly.

(a) [10 points] Show that $\lambda_y^{\log}(\hat{y})$ is 1-Lipschitz and convex for $y \in \{-1, 1\}$.

Answer: TODO

(b) [10 points] Show that for any 1-Lipschitz convex set of functions λ_y, the corresponding $L_S(w)$ is a convex, R-Lipschitz function of w.

Hint: $\|g \circ h\|_{\mathrm{Lip}} \le \|g\|_{\mathrm{Lip}} \|h\|_{\mathrm{Lip}}$, *since* $\|g(h(x)) - g(h(y))\| \le \|g\|_{\mathrm{Lip}} \|h(x) - h(y)\| \le \|g\|_{\mathrm{Lip}} \|h\|_{\mathrm{Lip}} \|x - y\|$.

Answer: TODO

Let $A_\lambda(S) = \arg\min_{w \in \mathbb{R}^d} L_S(w) + \lambda \|w\|^2$. We then have the bounds, from corollaries 13.8-13.9 of SSBD:

$$\begin{aligned} &\text{for any } w^* \in \mathbb{R}^d, && \mathop{\mathbb{E}}_S L_\mathcal{D}(A_\lambda(S)) \le L_\mathcal{D}(w^*) + \lambda \|w^*\| + \frac{2R^2}{\lambda n}; \\ &\text{if we use } \lambda = \frac{R}{B}\sqrt{\frac{2}{n}}, && \mathop{\mathbb{E}}_S L_\mathcal{D}(A_\lambda(S)) \le \inf_{w : \|w\| \le B} L_\mathcal{D}(w) + BR\sqrt{\frac{8}{n}}. \end{aligned} \tag{†}$$

It turns out the problem is also Convex-Smooth-Bounded, but the resulting bound is usually worse and also more annoying to work with.

(c) [10 points] Let $A'_B(S) \in \arg\min_{w \in \mathbb{R}^d : \|w\| \le B} L_S(w)$ (constrained ERM). Use Rademacher complexity to find a bound of the form

$$\mathbb{E}\, L_\mathcal{D}(A'_B(S)) \le \inf_{w : \|w\| \le B} L_\mathcal{D}(w) + Q$$

where Q is some term depending only on B, R, and n. Compare the resulting bound to (†).

Answer: TODO

For [no points (but hopefully a sense of personal satisfaction)], convince yourself that everything in this problem works for kernel logistic regression, i.e. $\mathcal{X} = \{x : \sqrt{k(x, x)} \le R\}$, $\ell^{\log}(h, (x, y)) = \lambda_y^{\log}(h(x))$, $A_\lambda(S) = \arg\min_{h \in \mathcal{F}} L_S^{\log}(h) + \lambda \|h\|_\mathcal{F}^2$, and $A'_B(S) = \arg\min_{h \in \mathcal{F} : \|h\|_\mathcal{F} \le B} L_S^{\log}(h)$. *Don't hand anything in for this, but if you want, it's nice to verify for yourself that the loss is still convex and R-Lipschitz, that* $A_\lambda(S)$ *is still stable, and* $A'_B(S)$ *still has the same Rademacher bound.*

3 A really hard Convex-Lipschitz-Bounded problem [15 points]

Convex-Lipschitz-Bounded problems are solvable by regularized loss minimization, which we showed gradient descent can approximately solve in polynomially many gradient steps. But this *doesn't* guarantee that Convex-Lipschitz-Bounded problems can be efficiently learned.

Let $\mathcal{H} = [0, 1]$ – nice and simple – but let the example domain $\mathcal{Z}$ be the set of all pairs of Turing machines T with input strings s. Define

$$\ell(h, (T, s)) = \begin{cases} \mathbb{1}(T \text{ halts on the input } s) & \text{if } h = 0 \\ \mathbb{1}(T \text{ does not halt on the input } s) & \text{if } h = 1 \\ h\ell(1, (T, s)) + (1 - h)\ell(0, (T, s)) & \text{if } 0 < h < 1. \end{cases}$$

Prove that this problem is Convex-Lipschitz-Bounded, but no computable algorithm can learn this problem.

Hint: If you have no idea what I'm talking about: look up the "halting problem."

Answer: TODO

4 Learning without concentration [25 points]

We're going to do an unsupervised learning task, where we try to estimate the mean of a distribution, but we do it with some *missing* observations. Specifically, let $\mathcal{B}$ be the unit ball $\mathcal{B} = \{w \in \mathbb{R}^d : \|w\| \le 1\}$, and let the samples be in $\mathcal{Z} = \mathcal{B} \times \{0,1\}^d$, where an entry $z = (x, \alpha)$ with α is a binary "mask" vector indicating whether the given entry is missing. We want to estimate the mean ignoring the missing entries, i.e. $\mathcal{H} = \mathcal{B}$ and

$$\ell(w, (x, \alpha)) = \sum_{i=1}^{d} \begin{cases} 0 & \text{if } \alpha_i = 1 \\ (x_i - w_i)^2 & \text{if } \alpha_i = 0. \end{cases}$$

(a) [10 points] Show that regularized loss minimization can PAC-learn this problem with a sample complexity independent of d.

Hint: Appeal to (*) *to show PAC learning.*

Answer: TODO

(b) [10 points] Let $\mathcal{D}$ be a distribution where x is always the fixed vector 0, and α has its entries i.i.d. $\text{Unif}(\{0,1\}) = \text{Bernoulli}(1/2)$. Let $n_{\mathcal{D}}(\varepsilon, \delta)$ denote the sample complexity of uniform convergence for this $\mathcal{D}$, so that if $n \ge n_{\mathcal{D}}(\varepsilon, \delta)$, then

$$\Pr_{S \sim \mathcal{D}^n}\left(\sup_{w \in \mathcal{H}} L_{\mathcal{D}}(w) - L_S(w) \le \varepsilon \right) \ge 1 - \delta.$$

Show that, for some (small) fixed $\varepsilon > 0$ and $\delta > 0$, $n_{\mathcal{D}}(\varepsilon, \delta)$ increases with d.

Hint: Show that if d is large enough relative to n, you're likely to get at least one dimension j where $(\alpha_i)_j = 1$ for all *your observed samples $i \in [n]$.*

Answer: TODO

(c) [5 points] Use these two results to describe a problem where RLM is a PAC learner, but uniform convergence doesn't hold. Describe why this doesn't contradict the fundamental theorem of statistical learning.

Answer: TODO

5 Challenge: Lasso and stability [10 points]

On-average replace-one stability is not the only notion of stability (nor even the most common). Another useful version of stability is *uniform stability*, which guarantees that

$$\forall S, S' \text{ of size } n \text{ differing for only one point,} \quad \sup_{z \in \mathcal{Z}} |\ell(A(S'), z) - \ell(A(S), z)| \leq \gamma(n)$$

for some $\gamma(n)$ that goes to 0 as $n \to \infty$.

One advantage of uniform stability is that you can get high-probability bounds on $L_{\mathcal{D}}(A(S))$, not just expectation bounds. (There's a simple route via McDiarmid, or over the past few years there have been some breakthroughs that give potentially much better bounds based on more complex analyses.)

For $\|\cdot\|_2$-regularized linear models (including kernels), the proof we did to show a bound on the on-average replace-one stability in fact shows the same bound on the uniform stability, $\gamma(n) \leq \frac{2\rho^2}{\lambda n}$.

The Lasso algorithm is based on the square loss and a $\|w\|_1 = \sum_{j=1}^{d} |w_j|$ regularizer:

$$A_\lambda(S) \in \operatorname*{arg\,min}_{w \in \mathbb{R}^d} L_S(w) + \lambda \|w\|_1.$$

(If there are multiple minimizers, let's have A_λ return an arbitrary one.) The Lasso algorithm is nice because it often returns sparse solutions, i.e. w with many $w_j = 0$.

Let's use $\mathcal{Z} = \mathcal{X} \times \mathcal{Y} = \{x \in \mathbb{R}^d : \|x\| \leq R\} \times [-M, M]$ for simplicity.

(a) [5 points] Show that the Lasso algorithm is not uniformly stable.

Hint: There's a reason I mentioned multiple minimizers above.

Answer: TODO

(b) [5 points] Show that the Lasso algorithm for any $\lambda > 0$ *is* on-average replace-one stable.

Hint: This is probably easiest to show with an "indirect" route via Rademacher complexity, as in Question 2 part (c). You'll also have to go through Lagrange duality and bound the Rademacher complexity of an appropriate set.

Answer: TODO