I’m likely to re-use at least some problems from year to year, so if you’re currently in or likely to take a future version of the course, please do not look at this solutions file.

Personally, I kind of understand the motivation to cheat sometimes in an undergrad course (although, obviously, please please don’t do it in undergrad courses, but like, I get why people would). But in a grad course…it’s really just not going to help you.

If you’re feeling super stressed / whatever and just need to get the assignment in, please write to me about an extension or re-weighting or some kind of route to making things work for you – I’m very willing to be flexible with this kind of thing in grad courses.
1 ERM vs SRM vs RLM [25 points]

Let’s say we’re given a training set $S = \{(x_1, y_1), \ldots, (x_n, y_n)\} \sim \mathcal{D}^n$, with $x_i \in \mathbb{R}^d$ and $y_i \in [-1, 1]$. We’re going to try to learning a homogeneous linear function $h$ from $\mathcal{H} = \{ x \mapsto w^\top x : w \in \mathbb{R}^d \}$, using the loss function $\ell_{abs}(h, (x, y)) = |h(x) - y|$. Let’s use the notation $w_h$ to refer to the $w$ inside $h$, i.e. $h(x) = w_h^\top x$.

You can assume $n > d \geq 3$, and that $\Pr(\|x\| \leq R) = 1$ for some finite $R$.

(a) [5 points] Suppose we just run empirical risk minimization, $h_S^{ERM} = \arg\min_{h \in \mathcal{H}} L_S^{\text{abs}}(h)$. What can we say about the generalization gap $L_D^{\text{abs}}(h_S^{ERM}) - L_S^{\text{abs}}(h_S^{ERM})$ using a type of bound covered in class?

Answer: Norm-based bounds aren’t going to work: they’ll just give an infinite upper bound. VC-based, margin, stability bounds don’t apply here either. So… we can’t say anything with the techniques from class.

Instead, let’s break $\mathcal{H}$ into parts: $\mathcal{H}_1 = \{ x \mapsto w^\top x : \|w\| \leq a \}$, $\mathcal{H}_2 = \{ x \mapsto w^\top x : \|w\| \leq 2a \}$, etc, for a constant $a > 0$ to be determined later. Now, run SRM, using “weight” $6/(\pi^2 k^2)$ for $\mathcal{H}_k$ as in class.

(b) [5 points] Write the solution $h_S^{SRM}$ in the form $\arg\min_k f(L_S^{\text{abs}}(h), \|w_h\|, a, n, R, \delta)$, for some function $f$, where $w_h$ is the vector $w$ “inside” $h$. Don’t include terms explicitly depending on $\mathfrak{R}_n(\mathcal{H}_k)$ or similar; it should be an expression that would be straightforward to implement in code. You can use the ceil function $\lceil \cdot \rceil$ (rounding up) or floor $\lfloor \cdot \rfloor$ (rounding down), if you’d like.

Hint: When we discussed SRM before, we focused on 0-1 loss. But there’s nothing actually in the argument that requires it, as long as we have a uniform convergence bound on $\mathcal{H}_k$. You may want to just re-derive it from the framework of minimizing an upper bound on $L_D(h)$.

Answer: We know that $\mathfrak{R}_n(\mathcal{H}_k) \leq akR/\sqrt{n}$, and the loss function is 1-Lipschitz. It’s also bounded: $|y| \leq 1$, and $|h(x)| \leq \|w_h\||x| \leq akR$, so $|h(x) - y| \leq 1 + akR$. Thus, with probability at least $1 - \delta w_k$, we have

$$\sup_{h \in \mathcal{H}_k} L_D^{\text{abs}}(h) - L_S^{\text{abs}}(h) \leq \frac{2akR}{\sqrt{n}} + (1 + akR) \sqrt{\frac{1}{2n} \log \frac{\pi^2 k^2}{6\delta}}.$$  

We can union this over all $\mathcal{H}_k$. Noting that $h \in \mathcal{H}_k$ for $k = \lceil \|w_h\|/a \rceil$, with probability at least $1 - \delta$ it holds simultaneously for all $h \in \mathcal{H}$ that

$$L_D^{\text{abs}}(h) \leq L_S^{\text{abs}}(h) + \frac{1}{\sqrt{n}} \left[ 2Ra \left( \frac{\|w_h\|}{a} \right) + \left( 1 + Ra \left( \frac{\|w_h\|}{a} \right) \right) \sqrt{\log \left( \frac{\|w_h\|}{a} \right) + \frac{1}{2} \log \frac{\pi^2}{\delta} \right}. \tag{1}$$

SRM is then the minimizer of the right-hand-side of (1). Unfortunately, unlike in the 0-1 loss case, we can’t just drop the confidence term that depends on $\delta$, since it’s scaled by something depending on $\|w_h\|$.

(c) [5 points] What can we say about the generalization gap $L_D^{\text{abs}}(h_S^{SRM}) - L_S^{\text{abs}}(h_S^{SRM})$ using a type of bound covered in class? Again, this should be a closed-form expression in terms of $\|w_h\|$, $a$, $R$, $n$, and $\delta$.

Answer: Simply plug in $h_S^{SRM}$ to (1).

Now, the last of our candidate algorithms is regularized loss minimization:

$$h_S^{RLM} = \arg\min_{h \in \mathcal{H}} L_S^{\text{abs}}(h) + \lambda \|w_h\|^2.$$  

(d) [5 points] What can we say about the generalization gap $L_D^{\text{abs}}(h_S^{RLM}) - L_S^{\text{abs}}(h_S^{RLM})$ using a type of bound covered in class?
Answer: $L_D^{abs}$ is R-Lipschitz as a function of $w_h$, since

$$||w_h^T x - y| - |(w_h')^T x - y|| \leq ||x||\|w_h - w_h'||.$$ 

Thus, a stability bound (SSBD Corollary 13.6, or the result on lecture 13 slide 10 – though that added $\lambda/2$ instead of $\lambda$, so divide the result by two) implies that

$$\mathbb{E} [L_D^{abs}(h_{RLM}^S) - L_S^{abs}(h_{RLM}^S)] \leq \frac{2R^2}{\lambda n}.$$ 

We can turn that into a “low-probability bound,” if we’d prefer, with Question 3 part (a).

(e) [5 points] Can we pick $\lambda$ and $a$ (maybe depending on $n$) so that RLM looks roughly like SRM? (It’s okay to be a little approximate here, e.g. ignoring logarithmic terms; we’re just talking about motivations.)

Answer: If $a < \|w_h\|$ is small enough that the rounding doesn’t really matter, but not so small that $\log \frac{\|w_h\|}{a}$ is big, then the SRM equation (1) looks a lot like

$$\arg\min_{h \in H} L_S^{abs}(h) + \frac{2R}{\sqrt{n}}\|w_h\|.$$ 

This is different from the usual RLM regularizer using $\|w_h\|^2$! Note that redefining $H_k$ to bound $\|w_h\|^2$ wouldn’t change that; it’d just fiddle with the rounding. It’s close enough that it makes sense as a vague inspiration, but I don’t think there’s a way to make it agree more closely.
2 Logistic regression [25 points]

Let $\mathcal{X} = \{x \in \mathbb{R}^d : \|x\| \leq R\}$, and $\mathcal{H} = \{w \in \mathbb{R}^d : \|w\| \leq B\}$. We’ll learn a linear predictor based on logistic loss,

$$\ell(w, (x, y)) = \log(1 + \exp(-yw^\top x)).$$

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

(a) [9 points] Show that $L_S(w)$ is a convex function of $w$.

Answer: The function

$$g(t) = \log(1 + \exp(-t))$$

is a convex function on $\mathbb{R}$: we have

$$g'(t) = \frac{\exp(t)}{1 + \exp(t)} = \frac{1}{1 + \exp(-t)} \quad g''(t) = \frac{-1}{(1 + \exp(-t))^2}(-\exp(-t)) = \frac{1}{\exp(t) + 2 + \exp(-t)} > 0.$$

Thus its composition with a linear function $-yw^\top x$ is also convex.

Alternatively, you could directly take the Hessian of $\ell$, which is similar but more annoying. You’d get

$$\nabla_w \ell(w, (x, y)) = g'(-yw^\top x)\nabla_w (-yw^\top x) = \frac{1}{1 + \exp(yw^\top x)}(-yx)$$

$$\nabla_w^2 \ell(w, (x, y)) = g''(-yw^\top x)(-yx)(-yx)^\top = \frac{1}{\exp(-yw^\top x) + 2 + \exp(yw^\top x)} xx^\top \succeq 0.$$

This tells us that $L_S$, the weighted sum of convex functions, is also convex.

(b) [8 points] Show $L_S(w)$ is $\rho$-Lipschitz, and give a (reasonably tight) upper bound on $\rho$.

Answer: As above, we have that $\ell(w, (x, y)) = [g \circ (w \mapsto -yw^\top x)](w)$, so its Lipschitz constant is upper-bounded by the product of the Lipschitz constant of each function. From the derivative equation above, $g'(t) \leq 1$, and $w \mapsto -yw^\top x$ is $\|x\|$-Lipschitz, so $\ell(w, (x, y))$ is also $\|x\|$-Lipschitz.

In general, we have that $\|\sum_i \alpha_i f_i\|_{\text{Lip}} \leq \sum_i \alpha_i \|f_i\|_{\text{Lip}}$, so we know that $L_S$ is $(\frac{1}{n} \sum_i \|x_i\|)$-Lipschitz, and we know that $\|x_i\| \leq R$, giving us that $\rho \leq R$.

(c) [8 points] Show $L_S(w)$ is $\beta$-smooth, and give a (reasonably tight) upper bound on $\beta$.

Answer: We have

$$\|\nabla_w \ell(w, (x, y)) - \nabla_w \ell(w', (x, y))\| = \|yg'(-yx^\top w)x + yg'(-yx^\top w')x\|$$

$$= \|g'(-yx^\top w) - g'(-yx^\top w')\| \|x\|$$

$$\leq \|w \mapsto g'(-yx^\top w)\|_{\text{Lip}} \|w - w'\| \|x\|$$

$$\leq \|g'\|_{\text{Lip}} \|w \mapsto -yx^\top w\|_{\text{Lip}} \|x\| \|w - w'\|$$

$$= \left(\sup_t g''(t)\right) \|x\|^2 \|w - w'\|.$$

We know $g''(t) \leq \frac{1}{4}$, since $\exp(t) + \exp(-t) \geq 2$. Thus $\nabla_w \ell(w, (x, y))$ is $\frac{1}{4}R^2$-Lipschitz, so $w \mapsto \ell(w, (x, y))$ is $\frac{1}{4}R^2$-smooth, and so is $L_S$.

You could also directly use $\|\nabla^2_w L_S(w)\|_{\text{op}}$; this is fairly intuitive as a generalization of upper bounding the gradient norm as a bound on the Lipschitz constant of a scalar function (here’s a proof).

As $\|w\|$ is bounded, this means that logistic regression is both Convex-Lipschitz-Bounded and Convex-Smooth-Bounded.

---

1Let $h(t) = \exp(t) + \exp(-t)$, so that $h'(t) = \exp(t) - \exp(-t)$ and $h''(t) = h(t) > 0$; it’s minimized at $t = 0$, where $h(0) = 2$. 

From Bounded Expected Risk to Agnostic PAC Learning [25 points]

Our stability and SGD analyses mostly bounded only the expected risk; we’ll now show this implies PAC learning.

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

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

You can assume that $L_D(h) = \mathbb{E}_{z \sim D} \ell(h, z)$ for a loss $\ell(h, z)$ bounded in $[0, 1]$.

(a) [10 points] Show that for every $\delta \in (0, 1)$, if $n \geq n_{\mathcal{H}}(\varepsilon\delta)$, then with probability of at least $1 - \delta$ it holds that $L_D(A(S)) \leq \inf_{h \in \mathcal{H}} L_D(h) + \varepsilon$.

Hint: Observe that the random variable $L_D(A(S)) - \inf_{h \in \mathcal{H}} L_D(h)$ is nonnegative, and rely on Markov’s inequality.

Answer: We know that $L_D(A(S)) \geq \inf_{h \in \mathcal{H}} L_D(h)$, because $A$ is a proper learning algorithm and so $A(S) \in \mathcal{H}$. Thus, if $n \geq n_{\mathcal{H}}(\varepsilon)$, we know that with probability at least $1 - \delta$,

$$L_D(A(S)) - \inf_{h \in \mathcal{H}} L_D(h) \leq \frac{1}{\delta} \mathbb{E} \left[ L_D(A(S)) - \inf_{h \in \mathcal{H}} L_D(h) \right] \leq \frac{\varepsilon}{\delta}$$

and so the desired property holds if we take $\varepsilon \delta$ instead of $\varepsilon$. This shows PAC learning, but with a bad rate.

(b) [15 points] For every $\delta \in (0, 1)$, let

$$n_{\mathcal{H}}(\varepsilon, \delta) = n_{\mathcal{H}}(\frac{\varepsilon}{2}) \left\lceil \log_2 \frac{\delta}{\varepsilon^2} \right\rceil + 8 \left[ \frac{\log \frac{1}{\delta} + \log \left( \log_2 \frac{\delta}{\varepsilon^2} \right)}{\varepsilon^2} \right].$$

Suggest a procedure that agnostic PAC learns the problem with sample complexity of $n_{\mathcal{H}}(\varepsilon, \delta)$, assuming that the loss function is bounded by 1.

Hint: Let $k = \lceil \log_2 \left( \frac{\delta}{\varepsilon^2} \right) \rceil$. Divide the data into $k+1$ chunks, where each of the first $k$ chunks has at least $n_{\mathcal{H}}(\varepsilon/4)$ examples. Train the first $k$ chunks using $A$. Using part (a), argue that the probability that for all of these chunks we have $L_D(A(S)) > \inf_{h \in \mathcal{H}} L_D(h) + \varepsilon/2$ is at most $2^{-k} \leq \delta/2$. Finally, use the last chunk as a validation set.

Answer: Let’s follow the hint, and divide the data up into $k$ chunks of size $n_{\mathcal{H}}(\varepsilon/4)$, and one remainder. Let $h_i$ denote the outcome of running $A$ on the $i$th chunk. We’re going to then use the last chunk to pick the best of the $\{h_i\}_{i=1}^k$: we’ll show that the best of those is not too bad, and then picking the best of them on the last chunk is not too much worse.

Plugging in part (a) with $\delta = \frac{\varepsilon}{2}$, we know that $L_D(h_i) \leq \inf_{h \in \mathcal{H}} L_D(h) + \frac{\varepsilon}{2}$ with probability at least $\frac{1}{2}$. Thus, the probability that all the $k$ independent $h_i$ are at least $\frac{\varepsilon}{2}$-suboptimal is at most $2^{-k} \leq \delta/2$.

Thus, we have $\delta/2$ remaining probability “budget” for picking the best of the $\{h_i\}_{i=1}^k$. We can just run ERM on a finite hypothesis class set of size $k$, as studied way back in lecture 3 (SSBD corollary 4.6). We want to get error worse than $\varepsilon/2$ with probability $\delta/2$, to combine nicely with the previous part. The result shows us that we can do that as long as the final part is of size at least

$$\frac{2}{(\varepsilon/2)^2} \left[ \log 2 + \log k + \log \frac{2}{\delta} \right] = \frac{8}{\varepsilon^2} \left[ \log \left( \log_2 \frac{2}{\delta} \right) + \log \frac{4}{\delta} \right].$$

Combining the two parts, we’ve shown an algorithm that can agnostically PAC learn the problem with

$$n \geq k n_{\mathcal{H}}(\varepsilon/4) + \left[ \frac{8}{\varepsilon^2} \left( \log \log_2 \frac{2}{\delta} + \log \frac{4}{\delta} \right) \right]$$
samples, as desired.
4 Perceptrons [25 points]

Let \( S = ((x_1, y_1), \ldots, (x_n, y_n)) \in (\mathbb{R}^d \times \{\pm 1\})^n \). Assume that a

\[
w^* \in \arg \min_{w \in \mathbb{R}^d : \forall i \in [n], y_i w^T x_i \geq 1} \|w\|
\]

exists. Let \( R = \max_i \|x_i\| \), and let

\[
f(w) = \max_{i \in [n]} 1 - y_i w^T x_i.
\]

(a) [5 points] Show that \( \min_{w : \|w\| \leq \|w^*\|} f(w) = 0 \), and that any \( w \) for which \( f(w) < 1 \) achieves \( L^{n-1}_S(w) = 0 \).

Answer: Consider \( f(w^*) \): since \( y_i(w^*, x_i) \geq 1 \), this means that \( f(w^*) \leq 0 \). Thus \( \min_{w : \|w\| \leq \|w^*\|} f(w) \leq f(w^*) \leq 0 \).

Now, suppose that there is some \( w' \) with \( \|w'\| \leq \|w^*\| \) and \( f(w') < 0 \). Thus \( y_i(w', x_i) > 1 \) for all \( i \). If \( \|w'\| < \|w^*\| \), this would contradict the definition of \( w^* \). Thus the only possibility is that \( \|w'\| = \|w^*\| \). But notice that \( f(w) \) is a continuous function of \( w \), and so then there would be some \( \alpha \in (0, 1) \) for which \( f(\alpha w') < 0 \), again contradicting the definition of \( w^* \). Thus \( \min_{w : \|w\| \leq \|w^*\|} f(w) \leq f(w^*) = 0 \).

Now, note that \( f(w) < 1 \) means that \( y_i w^T x_i \geq 0 \) for all \( i \), which means that \( y_i = \text{sign}(w^T x_i) \). Thus the zero-one error must be zero on \( S \).

(b) [5 points] Show how to calculate a subgradient of \( f \).

Hint: Recall that, at points for which \( f \) is differentiable, the gradient is a valid subgradient. For points where it’s not, think about the structure of \( f \): try drawing a sketch with \( d = 1 \) and \( n = 2 \).

Answer: Recall the definition of a subgradient of \( f \) at \( w \) is any vector \( v \) such that

\[
f(u) \geq f(w) + (u - w)^T v,
\]

and that when \( f \) is differentiable at \( w \), the gradient is the only subgradient.

Let’s denote \( f_i(w) = 1 - y_i w^T x_i \), so that \( f(w) = \max_{i \in [n]} f_i(w) \). Let \( i \in \arg \max_{i \in [n]} 1 - y_i w^T x_i \); this may or may not be unique, but just pick one. Then \( v_i := -y_i x_i \) is a valid subgradient. This is because it’s the gradient of \( f_i \) at \( w \), and hence from convexity, for all \( u \),

\[
f_i(u) \geq f_i(w) + (u - w)^T v_i;
\]

in fact, that holds with equality, since it’s an affine function. But we also know that

\[
f(u) = \max_{i \in [n]} f_i(w) \geq f_i(u).
\]

Thus

\[
f(u) \geq f_i(w) + (u - w)^T v_i = f(w) + (u - w)^T v_i,
\]

the definition of a subgradient.

(c) [5 points] Describe subgradient descent on the function \( f \) (initializing from \( w = 0 \)). UPDATE: no need to analyze the algorithm.

Answer: We start at \( w = 0 \). To go from \( w \) to the next iteration \( w^+ \): we find a subgradient of \( f \), \(-y_i x_i \) for the data point \( i \) maximizing the misclassification \( 1 - y_i w^T x_i \), i.e. minimizing \( y_i w^T x_i \). We then replace \( w \) by \( w^+ = w - \eta y_i x_i \).
To analyze this algorithm, note that the loss function $f$ is $R$-Lipschitz: we can see this e.g. by noticing that the norm of the subgradients is at most $\| -y_i x_i \| = \| x_i \| \leq R$. We can then apply results for subgradient descent on Lipschitz losses: except we didn’t really cover these in class, and SSBD only covers the average iterate case which doesn’t really make sense. Apologies about that.

**(d)** [5 points] Compare the resulting algorithm (not its analysis anymore) to the Batch Perceptron algorithm given in section 9.1.2 of SSBD. (We didn’t discuss this in class, but just reading the algorithm block should be enough.)

Answer: This is a very similar algorithm.

The first difference is: in the Batch Perceptron, we just pick an arbitrary mistake to update. In this algorithm, we find the *biggest* mistake, one maximizing $1 - y_i w^T x_i$, and update that. This doesn’t break the proof of SSBD’s Theorem 9.1 in any way, since it allows for updating based on an arbitrary mistake.

The second difference is the presence of the step size parameter $\eta$. Picking $\eta = 1$ makes them the same. If we don’t, though, the only difference is a global scale on $w$: we’ll still get the same sequence of minimizers with any constant choice of $\eta$.

**(e)** [5 points] Suppose that the training set $S$ is such that the training instances are linearly separable with a margin of $\gamma$. Refine the bound of SSBD’s Theorem 9.1 to include $\gamma$.

Answer: If $S$ is separable with a margin $\gamma$, then there is a $w^*$ of norm $1/\gamma$ achieving $f(w^*) = 0$, just from the definition of $f(w)$: if $f(w^*) = 0$, then $\min_{i \in [n]} y_i \langle w^*, x_i \rangle = 1$, and the geometric margin of the predictor $w^*$ is exactly

$$\frac{1}{\|w^*\|} \min_{i \in [n]} y_i \langle w^*, x_i \rangle = \frac{1}{\|w^*\|}.$$ 

Thus, the quantity $B$ in SSBD’s Theorem 9.1 is just $1/\gamma$, implying that the Batch Perceptron algorithm terminates with a linear separator in at most $(R/\gamma)^2$ steps, where $R = \max_{i \in [n]} \|x_i\|$ is as before.