Homework 7
APPM 7400 Spr 2020 Theoretical ML

Due date: Friday, March 13, before 1 PM **Instructor**: Prof. Becker
Theme: Boosting

Instructions Collaboration with your fellow students is OK and in fact recommended, although direct copying is not allowed. The internet is allowed for basic tasks (e.g., looking up definitions on wikipedia) but it is not permissible to search for proofs or to *post* requests for help on forums such as `http://math.stackexchange.com/` or to look at solution manuals. Please write down the names of the students that you worked with.

An arbitrary subset of these questions will be graded.

Reading You are responsible for reading chapter 10 about "boosting" of Understanding Machine Learning by Shai Shalev-Shwartz and Shai Ben-David (2014, Cambridge University Press), as well as chapter 7 in Foundations of Machine Learning, 2nd edition, by Mehryar Mohri, Afshin Rostamizadeh, and Ameet Talwalkar.

Problem 1: Show that the VC dimension of decision stumps in d dimensions is bounded $\text{VCdim}(\mathcal{H}) \le c_1 + c_2 \log_2(d)$ for some constants c_1 and c_2. (Any value of constants work, but you might try to get $c_1 = 6 - 4\log_2(\ln(2)) \approx 8.12$ and $c_2 = 2$, or better). Note: Exercise 9.5 in Mohri et al. shows more generally that binary Decision Trees of k nodes in d dimensions have $\text{VCdim}(\mathcal{H}) = \Omega(k \log(d))$.

Hint: After picking a dimension, how many dichotomies can a stump achieve? It may help to think of the ERM algorithm we derived for decision stumps; also, decision stumps have nice graphical interpretations, especially for $d = 2$. Use this to bound the size of the growth function $\tau(m)$, and then do the usual trick: argue that if VCdim $\ge m$, then we must have $\tau(m) \ge 2^m$, and use Lemma A.2.

Problem 2: Bounding the VC dimension of AdaBoost. Let $\text{VCdim}(\mathcal{H}) = d$, and consider boosting the base class $\mathcal{H}$ by T rounds of AdaBoost. Then Shalev-Shwartz and Ben-David shows that the boosted hypothesis class has VC dimension bounded by (Lemma 10.3)

$$\text{VCdim}(\mathcal{H}_{\text{boosted}}) \le T(d+1)(3\log(T(d+1)) + 2) \quad \text{if} \quad T \ge 3, d \ge 3$$

or via eq (7.9) in Mohri et al.:

$$\text{VCdim}(\mathcal{H}_{\text{boosted}}) \le 2(d+1)(T+1)\log_2((T+1)e).$$

We can make tighter bounds by not doing approximations. The proof of Lemma 10.3 bounds the growth function

$$\tau(m) \le \left(\frac{em}{d}\right)^{dT} \left(\frac{em}{T}\right)^{T} \tag{1}$$

and then gives some simplifying lower bounds to this, and then argues that if this is to be at least as big as 2^m (so that VCdim is m), it leads to the bounds we have on d. As an alternative, we can *numerically* solve $\tau(m) = 2^m$ for m, and then rely on monotonicity (no need to prove that for this exercise) that this is the VC dimension. Let d be the VC dimension of decision stumps in $\mathbb{R}^8$ (or actually the bound on it we derived in problem 1), and letting $T = 3$, compute the m that solves $\tau(m) = 2^m$ (or better, take the logarithm of both side before solving), and use this to get a more exact bound to the VC dimension for 8-dimensional decision trees that are boosted 3 times. (We don't actually solve $\tau(m) = 2^m$ since we don't know $\tau(m)$ exactly, but use the bound in Eq. (1) as a proxy for $\tau(m)$). Use numerical rootfinding, such as `fzero` in Matlab or `scipy.optimize.root_scalar` in Python.

Finally, compare with the generic bounds from the textbooks.

Problem 3:

a) Using the VC dimension of 8-dimensional decision stumps boosted 3 times (as calculated in the previous problem), estimate how many iid samples $m = |S|$ are needed to ensure that $L_{\mathcal{D}}(h) \le \widehat{L}_S(h) + \epsilon$ with probability at least $1 - \delta$, for $\epsilon = \delta = 0.05$?

b) How large of a validation set S_V would we need so that $L_{\mathcal{D}}(h) \le \widehat{L}_{S_V}(h) + \epsilon$ with probability at least $1 - \delta$, for the same values of ϵ and δ as in the previous part of the problem? [Note: this holds regardless of whether we used an ERM or boosting or any other algorithm]

Problem 4: **Coding** Download the dataset from github, which consists of $m = 10^4$ training data $\mathbf{z}_i = (\mathbf{x}_i, y_i)$ for binary labels $y_i \in \{\pm 1\}$ and $\mathbf{x}_i \in \mathbb{R}^8$. [1]

a) Run the ERM for decision stumps on this; you do not need to implement this yourself. For Matlab, you can use the code I wrote for the class `CreateDecisionStumpERM` (available on github website); for Python, several students can work together and share an implementation, or use an existing implementation (e.g., Scikit-learn's `DecisionTreeClassifier(max_depth=1)`).

b) Write code to boost the decision stump ERM

c) Using the dataset given to you (which you are welcome to use as you wish; e.g., you can split it into training/validation/test datasets; or you can run cross-validation), using whichever methods you like (model selection techniques discussed in class; generalization bounds discussed in this homework), decide how many rounds of boosting you want to do, and estimate an error ϵ (no need to tell me what δ is, but you may wish to choose a value).

Your output for this problem should be your estimate (or overestimate) of $L_{\mathcal{D}}(h)$ and your classifier h. Call this *estimate* of $L_{\mathcal{D}}(h)$ to be $\widehat{L}_{\mathcal{D}}(h)$. The output of your classifier should be in the form of a comma-separated text file, with as many rows as round of boosting that you did, and each row in the form of $j \in [8], \theta \in \mathbb{R}, b \in \{\pm 1\}, w > 0$, like

```
5, 228.997, 1, 0.480
2, -499.999, -1, 0.614
2, 333360.459, -1, 0.371
```

where the classifier at that round is defined as $h(\mathbf{x}) = b \cdot \operatorname{sign}(\theta - x_j)$, and then the overall classifier is $h_{\text{boosted}}(\mathbf{x}) = \operatorname{sign}\left(\sum_{t=1}^{T} w_t h_t(\mathbf{x})\right)$. I will get your value of T implicitly by looking at the number of rows. **This CSV file should be uploaded via Canvas**; but also turn in written/printed work with your answers to the first 3 problems, as well as your estimate $\widehat{L}_{\mathcal{D}}(h)$.

Grading for this problem: I will run your classifier on a new set of data (generated from the same distribution $\mathcal{D}$ over $\mathcal{Z}$ that I used to make the training data), and evaluate your error. If your true loss is less than your predicted $\widehat{L}_{\mathcal{D}}(h)$, then you will receive 10 bonus points on your midterm (not to exceed 100%). (Note: I will also grade this homework as usual as well, so it is recommended you try this exercise even if you don't want the bonus points).

However, since you could just estimate $\widehat{L}_{\mathcal{D}}(h) = 1$ so that it's always an over-estimate of the true loss, in order to receive the bonus points, you also need

$$\widehat{L}_{\mathcal{D}}(h) \le (1.5) \cdot \min_{\text{other classmates' estimate } h'} \widehat{L}_{\mathcal{D}}(h')$$

(and to ensure that all the students don't collude, I will include my own estimator; also, naturally, if any of your classmates had an incorrect estimate $\widehat{L}_{\mathcal{D}}(h)$, meaning that it was not an actual upper bound for $L_{\mathcal{D}}(h)$, then it will be excluded).

[1] Available at github.com/stephenbeckr/ML-theory-class/tree/master/Code where there is a Matlab `.mat` file (version 7.0, so it can be loaded via `scipy.io.loadmat` in Python) that contains both X and $\mathbf{y}$, as well as CSV files if you prefer that (separate files for X and $\mathbf{y}$).