Homework 6
APPM 7400 Spr 2020 Theoretical ML

Due date: Friday, Feb 28, 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.

Comment AdaBoost is the classical boosting algorithm that boosts weak learners into strong learners, e.g., enables you to reduce the training error arbitrarily low. This homework explores other types of "boosting" ideas; in particular, instead of using boosting to lower the error, these problems will use boosting to lower the failure probability.

Problem 1: Exercise 10.1 in Shalev-Shwartz and Ben-David, slight typo fixes. Suppose $\mathcal{H}$ is a fixed hypothesis class, and A is an algorithm such that learns it with arbitrary accuracy, but only with probability $1-\delta_0$ (not with arbitrarily high probability), i.e., if $m \geq m_{\mathcal{H}}(\epsilon)$ then for every distribution $\mathcal{D}$, with probability at least $1-\delta_0$, $L_{\mathcal{D}}(A(S)) \leq \min_{h\in\mathcal{H}} L_{\mathcal{D}}(h) + \epsilon$.

Suggest a procedure that relies on A and learns $\mathcal{H}$ in the usual agnostic PAC learning model and has a sample complexity of

$$m_{\mathcal{H}}(\epsilon,\delta) \leq k \cdot m_{\mathcal{H}}(\epsilon) + \left\lceil \frac{2\log(4k/\delta)}{\epsilon^2} \right\rceil, \quad \text{where} \quad k = \left\lceil \frac{\log(\delta/2)}{\log(\delta_0)} \right\rceil.$$

Hint: Divide the data into $k+1$ chunks, where each of the first k chunks is of size $m_{\mathcal{H}}(\epsilon)$, and train each chunk using A. Argue that the probability that *all* of these k classifiers, $h_j = A(S^{(j)})$ for $j \in [k]$, fails to be good is at most $\delta_0^k = \delta/2$, i.e.,

$$\mathbb{P}\left(\bigwedge_{j=1}^{k} \left(L_{\mathcal{D}}(A(h_j)) > \min_{h\in\mathcal{H}} L_{\mathcal{D}}(h) + \epsilon \right) \right) < \delta_0^k.$$

Then use the last chunk (which has a different size), to choose from the k hypotheses h_j, $j \in [k]$, and apply Corollary 4.6.

Problem 2: Exercise 2.2.8 in Vershynin's High-Dimensional Probability (2018, Cambridge University Press). Suppose you have a randomized algorithm that computes a binary output, and the algorithm can either be right or wrong (for example, it could be an algorithm that applies a stochastic primality test). Suppose your algorithm does only slightly better than chance, so that it is incorrect with probability $\delta_0 = 1/2 - \gamma$ for some margin $\gamma > 0$. Show that you can boost the

probability of success arbitrarily high, to $1 - \delta$ for any $\delta \in (0, 1)$, by running the algorithm k times (for different random seeds) and taking the majority vote, provided

$$k \geq \left\lceil \frac{1}{\gamma^2} \log\left(\frac{1}{\delta}\right) \right\rceil .$$

Hint: Let X_j be an indicator variable that indicates whether the j^{th} run of the algorithm was successful or not, and use Hoeffding's inequality. In fact, you can also show that you only need $k \geq \left\lceil \frac{1}{2\gamma^2} \log\left(\frac{1}{\delta}\right) \right\rceil$ but you don't need to show that extra factor of 2 for your homework.