TTIC 31020: Intro to ML **Winter 2024**

Homework 7

Due: 6 p.m., February 20th, 2024

Note *You may discuss these problems in groups. However, you must write up your own solutions and mention the names of the people in your group. Also, please do mention any books, papers or other sources you refer to. We recommend that you typeset your solutions in LaTeX. Please submit your solutions as a PDF document on Canvas.*

Challenge and Optional Questions *Challenge questions are marked with* challenge *and Optional questions are marked with* optional *. You can get extra credit for solving* challenge *questions. You are not required to turn in* optional *questions, but we encourage you to solve them for your understanding. Course staff will help with* optional *questions but will prioritize queries on non-*optional *questions.*

---

1. **Feature Selection** In this problem we will consider linear prediction, $h_w(x) = \langle w, \phi(x) \rangle$, where we hope to be able to find a predictor with a small number of features. Denote $I = \text{supp}(w) = \{i | w[i] \neq 0\}$ and $\|w\|_0 = |I|$. We will consider the following feature selection methods:

   Optimal Feature Selection $w_k = \arg\min_{\|w\|_0 \leq k} L_S(w)$, for the desired number of features k. This will be our "gold standard" reference, but the problem is that it is generally computationally intractable without enumerating over all possible subsets of k features.

   Greedy Feature Selection We start with $I_0 = \varnothing$ and add one feature at a time greedily:

   $$I_{k+1} = \arg\min_{I} \min_{\text{supp}(w) \subseteq I} L_S(w) \quad \text{s.t. } I_k \subset I, |I| = k+1 \tag{1}$$

   ℓ_1 **norm relaxation** $w_B = \arg\min_{\|w\|_1 \leq B} L_S(w)$. To find k features we select[1] B such that $|\text{supp}(w_B)| = k$ and use $I_k = \text{supp}(w_B)$. We might then want to fit a predictor on this subset without a norm constraint, i.e. use $w_k = \arg\min_{\text{supp}(w) \subseteq I_k} L_S(k)$.

   Filter-by-Correlation In a "filter" approach[2] we first select features and then learn a predictor using them. We will filter the features according to their correlation with the label Y. That is, for each feature $\phi(x)[i]$ we will calculate its (empirical) correlation coefficient ρ_i with the label y, then select the k features with the highest $|\rho_i|$. Recall that the correlation coefficient between two random variables Y and Z is defined as $\mathbb{E}[(Y - \mathbb{E}[Y])(Z - \mathbb{E}[Z])] / \sqrt{\mathbb{E}[(Y - \mathbb{E}[Y])^2]\mathbb{E}[(Z - \mathbb{E}[Z])^2]}$ (and not ~~$\mathbb{E}[(Y - \mathbb{E}[Y])]\mathbb{E}[(Z - \mathbb{E}[Z])] / \sqrt{\mathbb{E}[(Y - \mathbb{E}[Y])^2]\mathbb{E}[(Z - \mathbb{E}[Z])^2]}$~~), where for the empirical correlation we replace the expectations with empirical averages.

   The use of $|\rho_i|$ for feature selection is justified since it correspond to the best squared error we can get using a linear predictor that only uses $\phi(x)[i]$ and a bias term (**important background review exercise:** verify the following equality):

   $$\min_{a,b \in \mathbb{R}} L_S^{\text{sq}}(x \mapsto (a\phi(x)[i] + b)) = \text{Var}[y](1 - \rho_i^2). \tag{2}$$

   Ordering the features from highest to lowest $|\rho_i|$ is thus the same as ordering them by how useful they are *individually* for linear prediction w.r.t. the squared loss.

[1] This might be tricky! There might be multiple B with different supports of the same size, or the size of the support might jump when we increase B and there might not be any B with support of a certain size. In the latter case, we take I_k to include some arbitrary subset of the features that were added all at once when we increase B.

[2] The term "filter" is usually used in contrast to "wrapper" approaches which access the learning rule (in our case, ERM on linear predictors) and try it out on different feature subsets, as in the Optimal and Greedy approaches above.

We will use the squared loss $\ell(h_w(x), y) = (h_w(x) - y)^2$ to make some of the calculations easier, and $\phi(x) = x$ with $x \in \mathbb{R}^{100}$. We will also ignore the estimation error, and consider $m \to \infty$. That is, we will actually replace the empirical error L_S with the population error $L_{\mathcal{D}}$ in the methods above (and the population correlation instead of the empirical correlation).

For the two distributions bellow, and each of the four methods above, describe how I_k would look like for different k (noting ties), the order features would be added to it, and the smallest k s.t. $L_{\mathcal{D}}(w_k) \leq 0.01$. Which of the three tractable methods, if any, would work as well as the optimal method?

(a) $y = \frac{3}{\sqrt{10}} x_{100} + \frac{1}{\sqrt{10}} x_1$, where $x \sim \mathcal{N}(0, \Sigma)$ with:

$$\Sigma = \begin{bmatrix} 1 & 0 & 0 & \dots & 0 \\ 0 & 1 & 0.9 & \dots & 0.9 \\ 0 & 0.9 & 1 & \dots & 0.9 \\ & & \vdots & & \\ 0 & 0.9 & 0.9 & \dots & 1 \end{bmatrix} \in \mathbb{R}^{100 \times 100}.$$

(b) $x_1 = z_1$, $y = z_2$, $x_2 = z_1 + 0.0001 \cdot z_2$, $x_i = z_i + 0.001 \cdot z_2$ for $i \in \{3, 4, \dots, 100\}$, where $z \sim \mathcal{N}(0, I)$.

2. **Boosting as Coordinate Descent**

In this question we refer to the AdaBoost algorithm as presented in class, and using the notation from the lecture slides.

(a) Show that $L_{D^{(t+1)}}(h_t) = 0.5$. That is, that the weight update is such that the hypothesis we just used is useless under the new weighting, and an entirely different hypothesis is needed.

We will now understand AdaBoost as coordinate descent method on the empirical exp-loss. The exp-loss is defined as:

$$\ell_{\exp}(h(x); y) = e^{-yh(x)}$$

and the empirical exp-loss is accordingly given by:

$$L_S^{\exp}(h) = \frac{1}{|S|} \sum_{(x,y) \in S} \ell_{\exp}(h(x); y).$$

As was discussed in class, we will think of AdaBoost as learning a linear predictor $h_w(x) = \langle w, \phi(x) \rangle$ over a feature space, where each coordinate of the feature space corresponds to a single "weak hypothesis". We will therefore index both the feature vector and the weight vector w by $h \in \mathcal{B}$, where $\phi(x)[h] = h(x) \in \{\pm 1\}$ is the value of weak hypothesis $h \in \mathcal{B}$ on x. **The coordinates of $\phi(x)$ are therefore ± 1 valued**. The coordinates of the weight vector $w^{(T)}$ after t iterations of AdaBoost are then given by:

$$w^{(T)}[h] = \sum_{t=1..T-1 \text{ s.t. } h_t = h} \alpha_t,$$

where h_t and α_t are as in the description of AdaBoost given in class.

(b) **Coordinate Selection** We would like to update a coordinate h such that $\partial L_S^{\exp}(h_w) / \partial w[h] < 0$, and is as small as possible, so that increasing $w[h]$ (we will only consider increasing, rather then decreasing coordinate values) will decrease $L_S^{\exp}(h_w)$. Show that at iteration t of AdaBoost,

$$\frac{\partial L_S^{\exp}(h_{w^{(t)}})}{\partial w[h]} \propto L_{D^{(t)}}^{01}(h) - \frac{1}{2}$$

That is, choosing the a weak hypothesis with small weighted error $L_{D^{(t)}}^{01}(h)$ that is less than half, corresponds to choosing a coordinate with a small negative (thus far from zero) partial derivative (Hint: show that $D_i^{(t)} \propto e^{-y_i h_{w^{(t)}}(x_i)}$).

Note: Weak learning is guaranteed to find h where $\left|L^{01}_{D^{(t)}}(h) - \frac{1}{2}\right| \geq \gamma$, which ensures we are using coordinates with significant enough partial derivative to make progress. Choosing the best possible predictor in the base class (i.e. performing ERM on the base class with respect to the weighted distribution) would correspond to exactly picking the coordinate with the most negative partial derivative.

(c) optional **Coordinate Update** After selecting a coordinate, say h_t, the AdaBoost update is

$$w^{(t+1)}[h_t] = w^{(t)}[h_t] + \frac{1}{2}\log(\frac{1}{\epsilon_t} - 1)$$

while $w^{(t+1)}[h] = w^{(t)}[h]$ for all over coordinates h. Show that this update corresponds to exact line search on the coordinate, i.e. it is equivalent to:

$$w^{(t+1)} = \arg \min_{w \text{s.t. } \forall_{h \neq h_t} w[h] = w^{(t)}[h]} L^{\exp}_S(h_w)$$

(Hint: write $w[h_t] = w^{(t)}[h_t] + \alpha$, take the derivative w.r.t. α and set it to zero)

3. optional **Implicit Regularization in Gradient Descent**

Consider the dataset $S = \{(x_1, y_1), \ldots, (x_m, y_m)\}$ and a feature map $\phi : \mathcal{X} \to \mathbb{R}^d$ such that $d > m$.

We consider the least squares linear regression problem that you already saw in Homework 6. Let $\Phi \in \mathbb{R}^{m \times d}$ whose i^{th} row is $\phi(x_i)$ and $y \in \mathbb{R}^m$ whose entries are the labels y_i s. For simplicity, assume that Φ has full row-rank, i.e. $\text{Rank}(\Phi) = m$.[3] Consider the following training objective

$$L_S(w) = \frac{1}{m}\|\Phi w - y\|^2.$$

Note that, when $d > m$, the objective $L_S(w)$ has multiple global minimizers w such that $L_S(w) = 0$.

Gradient Descent. Consider the gradient descent update rule that you already saw in Homework 6.

$$w^{(0)} = 0, \qquad w^{(t+1)} = w^{(t)} - \eta \nabla_w L_S(w^{(t)}).$$

(a) Show that there is a unique w^* from $\text{Span}\{\phi(x_1), \ldots, \phi(x_m)\}$ such that $L_S(w^*) = 0$. Moreover, it is given by $w^* := \Phi^\top(\Phi\Phi^\top)^{-1}y$. Argue that w^* must also the unique minimum ℓ_2 norm zero training error solution from $\mathbb{R}^d$. In other words, show that

$$w^* = \arg \min_{w \in \mathbb{R}^d, L_S(w)=0} \|w\|_2.$$

(b) Show, by induction on t, that $w^{(t)}$ is always in $\text{Span}\{\phi(x_1), \ldots, \phi(x_m)\}$. Moreover, since $L_S(w)$ is convex in w, for a tuned step-size η, the algorithm converges to a global minimizer w^{GD} such that $L_S(w^{\text{GD}}) = 0$. Conclude that $w^{GD} = w^*$ and hence the gradient descent converges to the minimum ℓ_2 norm interpolating solution.

4. optional **Boosting the Confidence** In lectures, we mostly discussed the expectation $\mathbb{E}_{S \sim \mathcal{D}^m}[L_{\mathcal{D}}(A(S))]$, over a random training set $S \sim \mathcal{D}^m$, of the expected (population) error of a the predictor $A(S)$ returns by a learning rule $A(\cdot)$. It's frequently useful to be able to say that not only is this expectation low, but that with high probability over $S \sim \mathcal{D}^m$, a learning rule will return a predictor with small population error.

Let $A(S)$ be a learning rule that ensures

$$\mathbb{E}_{S \sim \mathcal{D}^{m_0}}[L_{\mathcal{D}}(A(S))] \leq \epsilon_0 \tag{3}$$

for some m_0, ϵ_0, on some distribution $\mathcal{D}$. Our goal in this question is to use A to construct, for every $\epsilon, \delta > 0$, a rule $\tilde{A}$ that ensures

$$\mathbf{P}_{\tilde{S} \sim \mathcal{D}^{\tilde{m}}}\left(L_{\mathcal{D}}(\tilde{A}(\tilde{S})) \leq \epsilon_0 + \epsilon\right) \geq 1 - \delta \tag{4}$$

using $\tilde{m} = \text{poly}(m, \epsilon, \delta)$ samples.

[3] If rows are linearly dependent, then remove some rows (i.e. examples) such that the resultant Φ has only independent rows.

(a) Use Markov's inequality to ensure that for any ϵ and for some δ_0 we have

$$\mathbf{P}_{S\sim\mathcal{D}^m}\left(L_{\mathcal{D}}(A(S)) \leq \epsilon_0 + \epsilon/2\right) \geq 1 - \delta_0.$$

Express δ_0 as a function of ϵ_0 and ϵ.

So far we didn't change the learning rule. But the problem is that our probability of failure δ_0 might be very high, perhaps much higher than 0.5 if we want ϵ/ϵ_0 to be small. That is, the probability $1 - \delta_0$ that we will succeed in finding a predictor s.t. $L_{\mathcal{D}}(A(S)) \leq \epsilon_0 + \epsilon/2$ might be very low. Next, we will *boost* this probability to an arbitrarily high probability $1 - \delta < 1$ by running A multiple times on independent training sets.

(b) Consider running $A(S_i)$ on k *independent* and identically drawn training sets $S_i \sim$ i.i.d.$\mathcal{D}^m$, $i = 1..k$. Show that for any $\delta, \epsilon > 0$ there exists k such that with probability at least $1 - \delta/2$, at least one of the runs outputs a predictor with error $L_{\mathcal{D}}(A(S_i)) \leq \epsilon_0 + \epsilon/2$. That is: $\mathbf{P}\left(\min_{i=1..k} L_{\mathcal{D}}(A(S_i)) \leq \epsilon_0 + \epsilon/2\right) \geq 1 - \delta/2$. Write down k as a function of ϵ_0, ϵ and δ. Here and in the rest of the question, use asymptotic (big-O) notation to simplify the expression. (Hint: use the previous part; first write down k in terms of δ_0 and δ and then plug in the expression of δ_0 in terms of ϵ and ϵ_0).

We can therefor get several predictors, where at least one of them has small error. But we want to output only a single predictor. This can be done using another large enough independent subset of S used for validation.

(c) Consider using another independent sample $S_{\text{val}} \sim \mathcal{D}^{m_v}$ and selecting $\hat{i} = \arg\min_{i=1..k} L_{S_{\text{val}}}(A(S_i))$. Use Hoeffding's inequality and a union bound to show that for any $\delta, \epsilon > 0$, there is some m_v such that using k as above, $\mathbf{P}\left(L_{\mathcal{D}}(A(S_{\hat{i}})) \leq \epsilon_0 + \epsilon\right) \geq 1 - \delta$. Write down m_v as a function of ϵ, δ.

(d) Putting this all together, for any $\epsilon, \delta > 0$, describe a learning rule $\tilde{A}$ that uses a sample of some size $\tilde{m}$ (that depends on m, ϵ_0, ϵ and δ) and outputs a predictor satisfying (4). Write down $\tilde{m}$ as a function of $m_0, \epsilon_0, \epsilon$ and δ. (Hint: $\tilde{A}$ will split $\tilde{S}$ into $k + 1$ independent subsets).