

Homework 4

*Student Name: Andrew Choi**UID: 205348339*

- Feel free to talk to other students in the class when doing the homework. You should, however, write down your solution yourself. You also must indicate on each homework with whom you collaborated and cite any other sources you use including Internet sites.
- You will write your solution in LaTeX and submit the **pdf file through Gradescope**. You also need to submit the **zipped LaTeX files to CCLE**. We will grade your homework based on the final version of the pdf file submitted to Gradescope. We will not grade the zipped Latex files on CCLE. However, failure to submitting your LaTeX files to CCLE will incur 2 points penalty out of 100 points.
- The homework (both pdf and zipped Latex source files) is due at **1:59 PM before the class**.

1. Exercise 10.1

Following the provided hint and the fix posted on Piazza, we first divide the data into $k + 1$ chunks where the first k chunks are of size $m_{\mathcal{H}}(\epsilon/2)$. We then train on the first k chunks using algorithm A . Denote $\mathcal{H}_k = \{h_1, \dots, h_k\}$ as the set of the hypotheses produced by this training. We also argue that with probability at most $\delta_0^k \leq \delta/2$, we have that $L_{\mathcal{D}}(A(S)) > \min_{h \in \mathcal{H}} L_{\mathcal{D}}(h) + \epsilon$. Then, note that with probability at least $1 - \delta_0^k$, we have the following inequality.

$$\min_{h \in \mathcal{H}_k} L_{\mathcal{D}}(h) \leq \min_{h \in \mathcal{H}} L_{\mathcal{D}}(h) + \epsilon/2$$

We can then apply ERM using the hypothesis class \mathcal{H}_k and the last training chunk which leads to the usual agnostic PAC learning model. To satisfy the sample complexity described in the problem, this last training chunk should be of size $\lceil (2 \log(4k/\delta))/(\epsilon/2)^2 \rceil$. Denote the hypothesis returned by performing ERM on \mathcal{H}_k as \hat{h} . Then, using Corollary 4.6, we obtain that with probability at least $1 - \delta/2$, $L_{\mathcal{D}}(\hat{h}) \leq \min_{h \in \mathcal{H}_k} L_{\mathcal{D}}(h) + \epsilon/2$. Finally, we can now apply the union bound where we obtain that with probability at least $1 - \delta$,

$$\begin{aligned} L_{\mathcal{D}}(\hat{h}) &\leq \min_{h \in \mathcal{H}_k} L_{\mathcal{D}}(h) + \epsilon/2 \\ &\leq \min_{h \in \mathcal{H}} L_{\mathcal{D}}(h) + \epsilon \end{aligned}$$

Therefore, we can conclude that this procedure relies on A and learns using the usual agnostic PAC learning model.

2. Exercise 12.1

Consider the hypothesis class \mathcal{H} consisting of halfspaces in \mathbb{R}^d . Then consider a sample S consisting of one (x, y) pair where $x = e_1$ and $y = 1$. Now consider $w = -e_1$. Then, $w \bullet x = -1$ which results in $L_S(w) = 1$. Note that this satisfies the first condition where for any w' s.t. $\|w - w'\| \leq \epsilon$, we have $L_S(w) \leq L_S(w')$. This is because $L_S(w')$ must be 1 since $0 < \epsilon < 1$, otherwise if $L_S(w') = 0$, this would result in $\|w - w'\| > 1$ which breaks the constraint. Therefore, w is a local minimum of L_S .

Now we show that w satisfies the second condition. This is quite simple as we can see that $w^* = e_1$ results in $L_S(w^*) = 0$ which satisfies the second condition that $L_S(w^*) < L_S(w)$. Therefore, w is not a global minimum of L_S .

3. Exercise 12.2

For clarification, the hypothesis class is defined as $\mathcal{H} = \{w \in \mathbb{R}^d : \|w\| \leq B\}$ and the input domain is defined as $\mathcal{X} = \{x \in \mathbb{R}^d : \|x\| \leq B\}$.

First we shall prove that the resulting learning problem is convex-Lipschitz-bounded. As proven in Example 12.1, we know that the function $f(x) = \log(1 + \exp(x))$ is convex as $f'(x)$ is a monotonically nondecreasing function. Furthermore, as proven in Example 12.4, we know the $f(x)$ is 1-Lipschitz over \mathbb{R} as $|f'(x)| \leq 1$. Likewise, from Example 12.4, we know that the linear function defined by $g(w) = v \bullet w + b$ is $\|v\|$ -Lipschitz using the Cauchy-Schwartz inequality. Then, by leveraging Claim 12.7, we know that the composition of Lipschitz functions preserves Lipschitzness and therefore the learning problem is B -Lipschitz. As the norm of each hypothesis is bounded by B , we conclude that the learning problem is convex-Lipschitz-bounded with parameters B and B .

Now, we move onto proving that the problem is convex-smooth-bounded. The convexity was proven previously. Using Example 12.5, we know that $f(x)$ is $(1/4)$ -smooth as $|f''(x)| \leq 1/4$. Then, by leveraging Claim 12.9, we have that the learning problem is $B^2/4$ -smooth. Therefore, we conclude that the learning problem is convex-smooth-bounded with parameters $B^2/4$ and B .

4. Exercise 12.3

Fix some (x, y) where $x \in \mathcal{X}$ and $y \in \mathcal{Y}$. Let $w_1, w_2 \in \mathbb{R}^d$. Let ℓ_1 and ℓ_2 be $\ell(w_1, (x, y))$ and $\ell(w_2, (x, y))$, respectively. To prove that this learning problem is R -Lipschitz, we must show that $|\ell_1 - \ell_2| \leq R\|w_1 - w_2\|_2 \forall w_1, w_2 \in \mathbb{R}^d$. Note that if $y(w_1 \bullet x) \geq 1$ and $y(w_2 \bullet x) \geq 1$, then $|\ell_1 - \ell_2| = 0 \leq R\|w_1 - w_2\|_2$. Now we must prove that the following holds for when $y(w_1 \bullet x) < 1$ and $y(w_2 \bullet x) < 1$. First, we can get rid of the absolute value by assuming that $\ell_1 \geq \ell_2$. We can then prove the following.

$$\begin{aligned} |\ell_1 - \ell_2| &= \ell_1 - \ell_2 \\ &= 1 - y(w_1 \bullet x) - (1 - y(w_2 \bullet x)) \\ &= y((w_2 - w_1) \bullet x) \\ &\leq \|w_1 - w_2\| \|x\| \\ &\leq R\|w_1 - w_2\| \end{aligned}$$

Note that this also holds for situations in which only one of the following is true: $y(w_1 \bullet x) \geq 1$ or $y(w_2 \bullet x) \geq 1$. This covers all possible cases and therefore, the problem of learning halfspaces with hinge loss is R -Lipschitz.

5. Exercise 13.3

To prove Theorem 13.12, we can leverage Theorem 13.2 and the linearity of expectation.

Denote $A(S)$ as h and $\operatorname{argmin}_{h \in \mathcal{H}} L_{\mathcal{D}}$ as h^* . Then the following proof can be derived.

$$\begin{aligned}
\mathbb{E}_{S \sim \mathcal{D}^m} [L_{\mathcal{D}}(h) - L_{\mathcal{D}}(h^*)] &= \mathbb{E}_{S \sim \mathcal{D}^m} [L_{\mathcal{D}}(h) - L_S(h) + L_S(h) - L_{\mathcal{D}}(h^*)] \\
&= \mathbb{E}_{S \sim \mathcal{D}^m} [L_{\mathcal{D}}(h) - L_S(h)] + \mathbb{E}_{S \sim \mathcal{D}^m} [L_S(h) - L_{\mathcal{D}}(h^*)] \\
&= \mathbb{E}_{S \sim \mathcal{D}^m} [L_{\mathcal{D}}(h) - L_S(h)] + \mathbb{E}_{S \sim \mathcal{D}^m} [L_S(h) - L_S(h^*)] \\
&\leq \epsilon_1(m) + \epsilon_2(m)
\end{aligned}$$

6. Exercise 14.2

To prove Corollary 14.14, simply plug in the given values of η and T into the inequality provided by Theorem 14.13. Doing so we arrive at the following proof. Denote $L_{\mathcal{D}}(w^*)$ as $\min_{w \in \mathcal{H}} L_{\mathcal{D}}(w)$ and $\epsilon \in (0, 1)$.

$$\begin{aligned}
\mathbb{E}[L_{\mathcal{D}}(\bar{w})] &\leq \frac{1}{1 - \eta\beta} \left(L_{\mathcal{D}}(w^*) + \frac{\|w^*\|^2}{2\eta T} \right) \\
&= \frac{1}{1 - \frac{1}{(1+3)/\epsilon}} \left(L_{\mathcal{D}}(w^*) + \frac{\|w^*\|^2(1 + 3/\epsilon)\epsilon^2}{24B^2} \right) \\
&\leq L_{\mathcal{D}}(w^*) + \frac{\epsilon^2}{24} + \frac{3\epsilon}{24} \\
&\leq L_{\mathcal{D}}(w^*) + \frac{\epsilon}{24} + \frac{\epsilon}{8} \\
&\leq L_{\mathcal{D}}(w^*) + \epsilon
\end{aligned}$$

7. Exercise 14.3

First we show that any w for which $f(w) < 1$ separates the examples in S . This is true because if $f(w) < 1$, then this satisfies $1 - y_i(w \bullet x_i) < 1 \ \forall i \in [m]$ which implies all examples are separated. Furthermore, the equality $\min_{w: \|w\| \leq \|w^*\|} f(w) = 0$ holds since w^* is chosen to be of minimal norm and $f(w^*) \leq 0$.

The subgradient of f is simply $-y_i x_i$ where $i = \operatorname{argmax}_i (1 - y_i(w \bullet x_i))$. At startup, the subgradient descent algorithm would initialize w to be (0^d) and then update the weights at each iteration by $w^{(t+1)} = w^t + \alpha y_i x_i$ where i is defined in the subgradient definition above and the learning rate is expressed as $\alpha \in (0, 1)$. This algorithm should then converge after $\|w^*\|^2 R^2$ iterations.

The subgradient descent Perceptron algorithm (abbreviated SDP) and Batch Perceptron (abbreviated BP) algorithms are very similar with two noticeable differences. First, is that they differ in their method of choosing which training example to use for the update. Whereas BP chooses any arbitrary example (x_i, y_i) where $y_i(w^t \bullet x_i) \leq 0$, SDP chooses the example (x_i, y_i) which is the $\min_i y_i(w^t \bullet x_i)$. The second difference is how the update rule itself is implemented as SDP involves a learning rate α where $w^{t+1} = w^t + \alpha y_i x_i$.

8. For the coding assignment, I implemented both adaboost and adaboost2 algorithms.

The results for **sklearn AdaboostClassifier** are shown below.

- Training error: 0.11
- Testing error: 0.15

The difference between `adaboost` and `adaboost2` lies in the implementation of the ERM algorithm for the decision stump weak learner where `adaboost` uses the assumption that $b = 1$. In this case, the training error was very high.

The results for **single decision stump** ($b=1$) are shown below.

- Training error: 0.39
- Testing error: 0.44

The results for **adaboost** are shown below.

- Training error: 0.33
- Testing error: 0.31

By allowing $b \in \{-1, +1\}$, `adaboost2` was able to outperform `adaboost` significantly. It even performed better than the `sklearn AdaboostClassifier`

The results for **single decision stump** ($b \in \{-1, +1\}$) are shown below.

- Training error: 0.34
- Testing error: 0.46

The results for **adaboost2** are shown below.

- Training error: 0.09
- Testing error: 0.13