Introduction to Machine Learning **Spring Semester**

Homework 3: April 14, 2021

Due: April 28, 2021

Theory Questions

1. **(20 points) Perceptron Lower Bound.** Show that for any $0 < \gamma < 1$ there exists a number $d > 0$, vector $w^\star \in \mathbb{R}^d$ and a sequence of examples $(x_1, y_1), \ldots, (x_m, y_m)$ such that:

   (a) $\|x_i\| = 1$.

   (b) $\frac{y_i x_i \cdot w^\star}{\|w^\star\|} \geq \gamma$.

   (c) Perceptron makes $\left\lfloor \frac{1}{\gamma^2} \right\rfloor$ mistakes on the sequence.

   (Hint: Choose $m = d = \left\lfloor \frac{1}{\gamma^2} \right\rfloor$ and let $\{x_i\}_i$ be the standard basis of $\mathbb{R}^d$)

2. **(12 points) Halving Algorithm.** Denote by $\mathcal{A}_{Hal}$ the Halving algorithm you have seen in class. Let $d \geq 6$, $\mathcal{X} = \{1, \ldots, d\}$ and let $\mathcal{H} = \{h_{i,j} : 1 \leq i < j \leq d\}$ where

$$h_{i,j}(x) = \begin{cases} 1 & (x = i) \vee (x = j) \\ 0 & otherwise \end{cases}.$$

   Show that $M(\mathcal{A}_{Hal}, \mathcal{H}) = 2$.

   (Definition of mistake bound $M(\mathcal{A}, \mathcal{H})$: Let $\mathcal{H}$ be a hypothesis class and $\mathcal{A}$ an online algorithm. Given any sequence $S = (x_1, h^\star(x_1)), \ldots, (x_m, h^\star(x_m))$ where m is an integer and $h^\star \in \mathcal{H}$, let $M_\mathcal{A}(S)$ be the number of mistakes $\mathcal{A}$ makes on the sequence S. Then $M(\mathcal{A}, \mathcal{H}) = \sup_S M_\mathcal{A}(S)$).

3. **(12 points) Interval growth function.** The goal of this exercise is to compute the growth function of the interval hypothesis class $H = \{h_{a,b} : a < b\}$ where $h_{a,b}(x) = 1$ if $x \in [a, b]$ and 0 otherwise. In other words, your goal is to give an explicit expression to $\Pi_H(m) = \max_{C:|C|=m} |H_C|$ where H_C is the restriction of H on the set C.

4. **(8 points) Convex functions.** Let $f : \mathbb{R}^n \to \mathbb{R}$ a convex function. Given $A \in \mathbb{R}^{n\times n}$ and $b \in \mathbb{R}^n$. Show that, $g(x) = f(Ax + b)$ is convex.

5. **(12 points)** Let $\mathbf{p} = (p_1, \ldots, p_n)$ be a discrete distribution, with $\sum_{i=1}^n p_i = 1$ and $p_i \geq 0$ for $i = 1, \ldots, n$. The *entropy*, which measures the uncertainty of a distribution, is defined by:

$$H(\mathbf{p}) = -\sum_{i=1}^{n} p_i \log p_i$$

where we define $0 \log 0 = 0$. Use Lagrange multipliers to show that the uniform distribution has the largest entropy (Tip: Ignore the inequality constraints $p_i \geq 0$ and show that the solution satisfies them regardless).

6. **(16 points)** Let A be a positive definite matrix (PSD), and let $f(\mathbf{w}) = \mathbf{w}^T A \mathbf{w}$. Show that there exist a step size η such that $T \leq O(\log(1/\epsilon))$ iterations of gradient descent (with some arbitrary initial point $\mathbf{w}_1 \neq 0$) are sufficient to achieve ϵ-optimal solution. That is,

$$f(\mathbf{w}_T) \leq \epsilon.$$

Where the O-notation may hide constants that depend on $\|\mathbf{w}_1\|$ and on the eigenvalues of A. (Hint: show that one can choose η so that the update of GD is of the form $\mathbf{w}_{t+1} = B\mathbf{w}_t$, where B is a PSD with eigenvalues values strictly smaller than 1).

Remark: f belongs to a family called "strongly convex" and "smooth" functions. It can be shown that GD has a similar convergence rate for any strongly convex and smooth function, and this rate is much better than the $O(1/\epsilon^2)$ rate we proved in class.

Programming Assignment

Submission Guidelines:

- Download the file `skeleton_perceptron.py` from Moodle.
- In the file `skeleton_perceptron.py` there is an helper function. The function reads the examples labelled 0, 8 and returns them with 0-1 labels. If you are unable to read the MNIST data with the provided script, you can download the file from here: `https://github.com/amplab/datasciencesp14/blob/master/lab7/mldata/mnist-original.mat`.
- Your code should be written with Python 3.
- Your submission should include exactly one file: `perceptron.py`.

1. **(20 points) Perceptron.** Implement the Perceptron algorithm (in the file name `perceptron.py`). Do not forget to normalize the samples to have unit length (i.e., $\|x_i\| = 1$).

   (a) **(8 points)** Use only the first $n = 5, 10, 50, 100, 500, 1000, 5000$ samples as an input to Perceptron. For each n, run Perceptron 100 times, each time with a different random order of inputs, and calculate the accuracy of the classifier on the test set of each run. You should therefore have 100 accuracy measurement per n. Print a table showing, for each n, the mean accuracy across the 100 runs, as well as the 5% and 95% percentiles of the accuracies obtained.

   (b) **(4 points)** The weight vector w, returned by Percepton, can be viewed as a matrix of weights, with which we multiply each respective pixel in the input image. Run Perceptron on the entire training set, and show w, as a 28×28 image, for example with `imshow(reshape(image, (28, 28)), interpolation='nearest')`. Give an intuitive interpretation of this image.

   (c) **(4 points)** Calculate the accuracy of the classifier trained on the full training set, applied on the test set.

   (d) **(4 points)** Choose one (or two) of the samples in the test set that was misclassified by Perceptron (with the full training set) and show it as an image (show the unscaled images). Can you explain why it was misclassified?