Homework #2

TA in charge: Yu-Xun Ruan

RELEASE DATE: 09/27/2010

DUE DATE: 10/11/2010, 4:00 pm IN CLASS

TA SESSION: 10/07/2010, 6:00 pm IN R110

Unless granted by the instructor in advance, you must turn in a hard copy of your solutions (without the source code) for all problems. For problems marked with (*), please follow the guidelines on the course website and upload your source code to designated places.

Any form of cheating, lying, or plagiarism will not be tolerated. Students can get zero scores and/or fail the class and/or be kicked out of school and/or receive other punishments for those kinds of misconducts.

Discussions on course materials and homework solutions are encouraged. But you should write the final solutions alone and understand them fully. Books, notes, and Internet resources can be consulted, but not copied from.

Since everyone needs to write the final solutions alone, there is absolutely no need to lend your homework solutions and/or source codes to your classmates at any time. In order to maximize the level of fairness in this class, lending and borrowing homework solutions are both regarded as dishonest behaviors and will be punished according to the honesty policy.

You should write your solutions in English with the common math notations introduced in class or in the problems. We do not accept solutions written in any other languages.

2.1 Simplified No-Free-Lunch Theorem

Let $\mathcal{X} = \{\mathbf{x}_1, \mathbf{x}_2, \ldots, \mathbf{x}_N, \mathbf{x}_{N+1}, \ldots, \mathbf{x}_{N+M}\}$ and $\mathcal{Y} = \{-1, +1\}$ (binary classification). Here the set of training examples is $\mathcal{D} = \left\{(\mathbf{x}_n, y_n)\right\}_{n=1}^{N}$, where $y_n \in \mathcal{Y}$, and the set of test inputs is $\left\{\mathbf{x}_{N+m}\right\}_{m=1}^{M}$. The *Off-Training-Set error* (*OTS*) with respect to an underlying target f and a hypothesis g is

$$E_{OTS}(g, f) = \frac{1}{M} \sum_{m=1}^{M} [\![g(\mathbf{x}_{N+m}) \neq f(\mathbf{x}_{N+m})]\!] .$$

(1) (5%) Say $f(\mathbf{x}) = 1$ for all $\mathbf{x}$ and $g(\mathbf{x}) = \begin{cases} 1, & \text{for } \mathbf{x} = \mathbf{x}_k \text{ and } k \text{ is even and } 1 \leq k \leq M+N \\ -1, & \text{otherwise} \end{cases}$.
What is $E_{OTS}(g, f)$?

(2) (5%) We say that a target function f can "generate" $\mathcal{D}$ in a noiseless setting if $f(\mathbf{x}_n) = y_n$ for all $(\mathbf{x}_n, y_n) \in \mathcal{D}$. For all possible $f : \mathcal{X} \to \mathcal{Y}$, how many of them can generate $\mathcal{D}$ in a noiseless setting?

(3) (5%) For a fixed g, if all those f in (2) are equally likely in probability, what is the expected off-training-set error $\mathbb{E}_f\{E_{OTS}(g, f)\}$?

(4) (5%) A determistic algorithm A is defined as a procedure that takes $\mathcal{D}$ as an input, and outputs a hypothesis g. Argue that for any two deterministic algorithms A_1 and A_2,

$$\mathbb{E}_f\Big\{E_{OTS}\big(A_1(\mathcal{D}), f\big)\Big\} = \mathbb{E}_f\Big\{E_{OTS}\big(A_2(\mathcal{D}), f\big)\Big\}.$$

You have now proved that "in a noiseless setting (f generates $\mathcal{D}$), for a fixed $\mathcal{D}$, if all possible f are equally likely, any two deterministic algorithms are the same in terms of $\mathbb{E}_f\{E_{OTS}\}$."

2.2 Marbles and Coins

(1) (5%) Do Exercise 1.9 of LFD.

(2) (5%) Do Exercise 1.10 of LFD.

(3) (5%) Do Exercise 1.11-1 of LFD.

(4) (5%) Do Exercise 1.11-2 of LFD.

(5) (5%) Do Exercise 1.11-3 of LFD.

2.3 Learning Games

(1) (5%) Do Exercise 1.12-1 of LFD.

(2) (5%) Do Exercise 1.12-2 of LFD.

(3) (5%) Do Exercise 1.12-3 of LFD.

(4) (5%) Do Exercise 1.12-4 of LFD.

2.4 Probably Approximately Correct

Read the derivation that links Equation (1.6) to Equation (2.1) on LFD Page **2**-2. In particular, let

$$\epsilon(M, N, \delta) = \sqrt{\frac{1}{2N} \ln \frac{2M}{\delta}}.$$

(1) (5%) Take $\delta = 0.03$ and $M = 1$, how many examples do we need to make $\epsilon(M, N, \delta) \leq 0.05$?

(2) (5%) Take $\delta = 0.03$ and $M = 100$, how many examples do we need to make $\epsilon(M, N, \delta) \leq 0.05$?

(3) (5%) Take $\delta = 0.03$ and $M = 10000$, how many examples do we need to make $\epsilon(M, N, \delta) \leq 0.05$?

The title of this problem, *Probably Approximately Correct*, states what we can interpret from (2.1) if we have enough training examples. "Probably" means the statement is true with a high probability ($\geq 1 - \delta$). "Approximately" means that every $E_{\text{out}}(g)$ is close to $E_{\text{in}}(g)$ (within ϵ). "Correct" means that we can guarantee $E_{\text{out}}(g)$ to be small (by getting some decision function g with small $E_{\text{in}}(g)$).

2.5 Adaptive Perceptron Learning (*)

We know that the perceptron learning rule works like this: In each iteration, pick a random $(\mathbf{x}^{(t)}, y^{(t)})$ and compute $\rho^{(t)} = \mathbf{w}^{(t)} \bullet \mathbf{x}^{(t)}$. If $y^{(t)} \cdot \rho^{(t)} \leq 0$, update $\mathbf{w}$ by

$$\mathbf{w}^{(t+1)} \longleftarrow \mathbf{w}^{(t)} + y^{(t)} \cdot \mathbf{x}^{(t)} \ ;$$

One may argue that the algorithm did not take the "closeness" between $\rho^{(t)}$ and $y^{(t)}$ into consideration. Let's look at another perceptron learning algorithm: In each iteration, pick a random $(\mathbf{x}^{(t)}, y^{(t)})$ and compute $\rho^{(t)} = \mathbf{w}^{(t)} \bullet \mathbf{x}^{(t)}$. If $y^{(t)} \cdot \rho^{(t)} \leq 1$, update $\mathbf{w}$ by

$$\mathbf{w}^{(t+1)} \longleftarrow \mathbf{w}^{(t)} + \eta \cdot \left(y^{(t)} - \rho^{(t)}\right) \cdot \mathbf{x}^{(t)} \ ,$$

where η is some constant. That is, if $\rho^{(t)}$ agrees with $y^{(t)}$ a lot (their product is > 1), the algorithm does nothing. On the other hand, if $\rho^{(t)}$ is further from $y^{(t)}$, the algorithm changes $\mathbf{w}^{(t)}$ more. In this problem, you are asked to implement this algorithm and check its performance.

(1) (5%) Generate a training data set of size 100 as directed in Homework Problem 1.3. Generate a test data set of size 10000 from the same process. Run the algorithm above with $\eta = 100$ on the training data set until it converges (no more possible updates) or a maximum of 1000 updates has been reached to get g. Plot the training data set, the target function f, and the final hypothesis g on the same figure. Estimate the out-of-sample error with the test set.

(2) (5%) Use the data set in (1) and redo everything with $\eta = 1$.

(3) (5%) Use the data set in (1) and redo everything with $\eta = 0.01$.

(4) (5%) Compare the results that you get from (1) to (3).

The algorithm above is a variant of the so-called Adaline (*Adaptive Linear Neuron*) algorithm for perceptron learning.