582669 Supervised Machine Learning (Spring 2011)
Homework 3 (10 February)

Turn this homework in no later than **Tuesday, 8 February, at 15:00**.

1. Suppose we know for some online prediction algorithm in the expert setting a loss bound

$$L_A \leq \left(1 + \frac{\eta}{4}\right) L_* + \frac{\ln n}{\eta},$$

where L_A is the loss of the algorithm with learning rate η, L_* is the loss of the best expert and n is the number of experts. Assume we know a bound K such that $L_* \leq K$. Show how we can choose a learning rate that depends only on K and n and results in a regret bound

$$L_A - L_* \leq a\sqrt{L_* \ln n}$$

for some $a > 0$. What value a you can get, and what is the learning rate you use for that?

Hint: The main ingredients are

- writing the bound as $L_A - L_* \leq g(\eta, L_*, n)$ for some g and
- minimising $g(\eta, K, n)$ as a function of η.

Remark: This is the bound we would get by considering Corollary 2.8 from the lectures in the limit of small η and dropping terms that are $\Theta(\eta^2)$ or smaller. Thus, the bound is not exactly satisfied by the Aggregating Algorithm. Nevertheless this problem shows the basic idea leading to the square root style regret bounds. The proof of the actual Aggregating Algorithm bound stated in Theorem 2.9 is given by Cesa-Bianchi et al., *How to use expert advice*, Theorem 4.4.3, and a bit complicated.

2. (a) Show that the arithmetic mean of non-negative numbers $a_1, \ldots, a_n$ is at least their geometric mean, i.e.,

$$\frac{1}{n}\sum_{i=1}^{n} a_i \geq \left(\prod_{i=1}^{n} a_i\right)^{1/n}.$$

   (b) Given $\boldsymbol{p} \in \mathbb{R}^n$ and $\boldsymbol{q} \in \mathbb{R}^n$ such that $p_i \geq 0$ and $q_i \geq 0$ for all i and $\sum_{i=1}^n p_i = \sum_{i=1}^n q_i = 1$, we define their *relative entropy* (or *Kullback-Leibler divergence*) as

$$d_{\mathrm{KL}}(\boldsymbol{p}, \boldsymbol{q}) = \sum_{i=1}^{n} p_i \ln \frac{p_i}{q_i}.$$

   Show that

$$d_{\mathrm{KL}}(\boldsymbol{p}, \boldsymbol{q}) \geq 0$$

   holds for all $\boldsymbol{p}$ and $\boldsymbol{q}$ (that satisfy the conditions above). This is known as the *information inequality*.

Hint: Both of these can be proven by applying Jensen's inequality to the convex function $f(x) = -\ln(x)$. In other words, you can write the inequalities as

$$-\sum_i v_i \ln x_i \geq -\ln \sum_i v_i x_i$$

for suitable chosen v_i and x_i.

More problems on the next page!

3. Consider the following *Muroga's function* defined for $\boldsymbol{x} \in \{-1, 1\}^d$:

   if $x_1 = 1$ **then return** 1
   else if $x_2 = 1$ **then return** -1
   else if $x_3 = 1$ **then return** 1
   ...
   else if $x_d = 1$ **then return** 1
   else return -1.

   (This is for odd d, for even d the signs at the end go the other way). Show that Muroga's function can be represented as a linear classifier (you may use a threshold term if you wish). What margin do you get?

   You do not need to show that your construction is optimal, but **for extra credit** (worth one regular problem) you may prove that the largest possible margin is exponentially small in d.

4. Consider the linear classifier $f(\boldsymbol{x}) = \text{sign}(\boldsymbol{w} \cdot \boldsymbol{x})$ for $\boldsymbol{x} \in \mathbb{R}^d$ where $w_1 = w_2 = 1$ and $w_i = 0$ for $i = 3, \ldots, d$.

   We generate a random sample as follows. First, we draw a large number of instances $\boldsymbol{x}_t$ from the uniform distribution over the cube $[-1, 1]^d$. Then we classify the instances using the above classifier f. Finally, we discard from the sample the points where the margin is below some value γ we decide in advance. Therefore we get a sample that is linearly separable with margin γ by the classifier f.

   Implement the sampling method and the Perceptron algorithm. Study how the number of mistakes made by the algorithm changes when you

   - keep dimension d fixed but let the margin γ vary
   - keep the margin γ fixed but let dimension d vary.

   Is the behaviour of the algorithm similar to what you would expect from the Perceptron Convergence Theorem?

   Your solution should consist of a brief explanation of the observations you made, a couple of representative plots to support this, and a printout of your program code.