Problem Set 2

1. [Regret bounds]

In the experts framework we compared an online algorithm that was allowed to pick among n (expert) strategies, with the best choice of a single expert in hindsight. In this question we first introduce a different measure of performance of the online algorithm, called regret. The regret is how much worse the online player does than the best offline player. i.e. if the best expert suffers a loss of L while the expected loss of the online player is $E[L(A)]$, then the regret is $E[L(A)] - L$.

Assume that the game lasts for T rounds, and the loss in each round is in the interval $[0, 1]$.

(a) Show that the regret suffered by the experts algorithm (with appropriate parameters) compared to the best expert is $O(\sqrt{T \log n})$.

(b) How important was it that the online player was allowed to switch between experts, while the offline player had to stick to a single expert? To answer let us consider the regret suffered by the online player compared to an offline player who is allowed to switch between experts at every step. Construct an example where L^*, the loss suffered by the offline player is 0, while the expected regret of A is at least $T(1 - \frac{1}{n})$.

2. [Linear Programming/Duality]

(a) Recall that the congestion minimization problem is the following: Given a graph $G = (V, E)$, route one unit of flow between k pairs of vertices, $(s_1, t_1), \ldots, (s_k, t_k)$.

The size of the linear program for the congestion minimization problem presented in class had an exponential dependence on the size of the graph. This was because the program had a variable corresponding to each path between any of the k pairs.

Give a polynomial sized linear program for the Congestion minimization problem. (Hint: the linear program should have a variable, $f_i(e)$, corresponding to originating at s_i and destined for t_i that goes through edge e.)

(b) Take its dual, and argue that it corresponds to the toll problem from class.

3. [Kernel Trick and Perceptrons]

In this problem, we will demonstrate the "Kernel trick" with a small example. The setting is that we have a set of $\pm$ labeled points S belonging to R^n. In the case that they are linearly separable, the perceptron algorithm finds a linear separator efficiently. Moreover, and this is the crucial observation for the Kernel trick, the algorithm's dependence on the points is limited to the values of dot products of points in S.

Consider the more general case where the points are not linearly separable but say separable by a quadratic surface. e.g. the point $x = (x_1, \ldots, x_n) \in R^n$ is labeled $+$ if and only if it satisfies the inequality $\sum_{i,j} a_{i,j} x_i x_j \leq b$. Of course, if instead of giving the point $x = (x_1, \ldots, x_n)$, we give the expanded n^2 dimensional point y whose components are $x_i x_j$ for all i, j, there is a linear separator. However, this reduction may be expensive as the size of these vectors is quite large. In fact, we can implicitly rely on such a representation in the perceptron algorithm without incurring this overhead because it only needs the result of dot products in the expanded representation. The idea is to compute the dot product in the expanded representation as a function of the dot products in the original representation. This places some constraints on the expanded representation.

We will demonstrate the framework in a low dimensional example.

Let S be a set of labelled points in R^2, where the label is determined by whether or not the point is within distance 1 of the origin.

(a) Consider the mapping, $\phi([x_1, x_2]) = [1, x_1, x_1, x_2, x_2, x_1^2, x_2^2, x_1 x_2, x_2 x_1]$. Show that for two vectors x and y, that $K(x, y) = (1 + x \cdot y)^2 = \phi(x) \cdot \phi(y)$.

(b) What are the weights in the expanded space of a linear separator that is sure to separate the labeled points?

(c) Assuming that the points are at least δ away from the circle in the original space, show that the margin in the expanded space of the linear separator is at least $\delta/2$ assuming $\delta \leq 1/2$. This calculation is not particularly precise.

(d) Using the margin bound above, state a reasonable upper bound on the number of mistakes the perceptron algorithm make for the set of points.

(e) How many computations of $K(x, y)$ does one need to predict labels in the percepton algorithm?

4. [Eigenvectors and Connectivity]

Show that a d-regular graph, G, is connected if and only iff $\lambda_2(M) < 1$, where M is the normalized adjacency matrix of G.

5. [Eigenvectors and the Hypercube]

(a) Recall that the "dimension" eigenvectors form an orthogonal basis for the eigenspace corresponding to the second largest eigenvalue of a hypercube. For coordinate 1 this is the vector that assigns $+1$ to every vertex $0x$ where $x \in \{0, 1\}^{d-1}$ and -1 to every vertex $1x$.

Now, consider the average of these d vectors. Describe this vector. There is a a natural cut defined by this vector: i.e. let S be the set of vertices assigned positive values by this vector. Describe the set S.

(b) Let x be an arbitrary vertex of the hypercube. We can define a set S with $2^d/2$ vertices as follows: let S be the set of vertices within distance $d/2$ of x. Show that for each such choice of S (there are 2^d), there is a vector in the second eigenspace such that the natural cut defined by that vector is $(S, \bar{S})$.