ENGN 2520 Homework 3
Due Friday March 9 by 4pm

Problem 1

Let H denote the set of axis-parallel rectangles in the plane. That is, each $h \in H$ is defined by a range of x values and a range of y values. Each rectangle defines a binary classifier that assigns label $+1$ to points inside the rectangle and label -1 to points outside the rectangle. Show that the VC dimension of H is 4.

Problem 2

Let X be an input space and H be a finite set of functions from X to $\{-1, +1\}$. Show that the VC dimension of H is at most $\log |H|$.

Problem 3

In this assignment you will implement a multiclass SVM to recognize handwritten digits. You will use the data from Homework 2 that is available on the course website.

A multiclass SVM learns functions from $\mathbb{R}^D$ to $\{1, \ldots, k\}$ where k is the number of classes. The classifier is defined by k weight vectors $w_1, \ldots, w_k \in \mathbb{R}^D$ using the classification rule

$$f(x) = \operatorname*{argmax}_y w_y^T x$$

where y ranges from 1 to k. We can think of $w_y^T x$ as a score of class y on example x, and the classifier selects the class with maximum score.

Let $T = \{(x_1, y_1), \ldots, (x_n, y_n)\}$ be a set of training examples. Here $x_i \in \mathbb{R}^D$ and $y_i \in \{1, \ldots, k\}$. We would like to learn weight vectors $w_1, \ldots, w_k$ such that

$$\operatorname*{argmax}_y w_y^T x_i = y_i.$$

To obtain a large-margin classifier we can look for weight vectors $w_1, \ldots, w_k$ that have low norms and such that the score of class y_i on x_i is above the score of all other classes

by at least 1. In analogy to the binary SVM objective function we can define an objective function for the multiclass case as follows

$$E(w_1, \ldots, w_k) = \sum_{j=1}^{k} ||w_j||^2 + C \sum_{i=1}^{n} L((w_1, \ldots, w_k), (x_i, y_i)).$$

where the loss of $(w_1, \ldots, w_k)$ on (x, y) is

$$L((w_1, \ldots, w_k), (x, y)) = \begin{cases} 0 & \text{if } w_y^T x > (\max_{\hat{y} \neq y} w_{\hat{y}}^T x) + 1 \\ (\max_{\hat{y} \neq y} w_{\hat{y}}^T x) + 1 - w_y^T x & \text{otherwise} \end{cases}$$

We can define subgradients of L as follows:
Let $\hat{y} = \text{argmax}_{\hat{y} \neq y} w_{\hat{y}}^T x$.
Let $w_{j,l}$ be the l-th entry in w_j.
If $w_y^T x \geq w_{\hat{y}}^T x + 1$ then

$$\frac{\partial L((w_1, \ldots, w_k), (x, y))}{\partial w_{j,l}} = 0$$

If $w_y^T x < w_{\hat{y}}^T x + 1$ and $j = y$ then

$$\frac{\partial L((w_1, \ldots, w_k), (x, y))}{\partial w_{j,l}} = -x_l$$

If $w_y^T x < w_{\hat{y}}^T x + 1$ and $j = \hat{y}$ then

$$\frac{\partial L((w_1, \ldots, w_k), (x, y))}{\partial w_{j,l}} = x_l$$

If $w_y^T x < w_{\hat{y}}^T x + 1$ and $j \neq y$ and $j \neq \hat{y}$ then

$$\frac{\partial L((w_1, \ldots, w_k), (x, y))}{\partial w_{j,l}} = 0$$

(a) Implement gradient descent for optimizing the multiclass SVM objective function. The input to the optimization function should be: the training set, the value of C, the step size s, and the number of gradient descent interations T.

For parts (b) and (c) you will need to experiment with different values of C, s and T. You should train models with different values of C (such as $C = 0.01, 0.1, 1, 10, 100$) and report the results that lead to the best performance at test time. For s and T you need to pick s small enough so that the objective function goes down consistently over time, but not too small so as to require too many steps of gradient descent. For a given choice of s and T you should look at a plot of the objective function over time. T should be large enough that in the end the plot is more or less flat. s should be small enough so that the objective is going down smoothly on average.

(b) Use gradient descent to train a multiclass SVM model for digit recognition. You should use the training data from Homework 2. Make a visualization of the model for each digit by drawing each vector w_y as a 28x28 image. The brightness of a pixel should depend on the value of the relevant entry in w_y. Note that w_y will have both positive and negative values and you need to take this into account to make a meaningful picture.

(c) Use the resulting classifier to classify the test data. What fraction of the test digits were correctly classified? Compute a 10x10 confusion matrix where entry (i,j) specifies how often digit i was classified as digit j.

You should turn in a writeup that includes

1) a derivation of the gradient for the whole objective function

2) a printout of your source code

3) a plot of E as a function of the gradient descent iteration during training

4) a visualization of the final models you obtained

5) the results from part (c)