1. (2 pts) Read the handout on the back-propagation algorithm. Write a single mathematical formula using the sigma function $\sigma()$, the weights (like $w_{53}$), and the inputs ($z_1$ and $z_2$) for the value produced by the network drawn in Figure 1 of the handout. Assume that the activation function on the output node is the identity function.

2. (3 pts) Consider the hypothesis class of intervals on the real line. (i.e. each instance $b \in \mathbb{R}$ and $H$ consists of all $h_{\ell,r}$ having the form $h_{\ell,r}(x) = +1$ if $\ell \leq x \leq r$ and $-1$ otherwise. (Note: whenever $r < \ell$ the hypothesis $h_{\ell,r}$ is the same "empty hypothesis", mapping everything to $-1$.)

Determine the VC-dimension of intervals on the real line. Recall that the VC-dimension is the size of the largest shattered set. To show the VC-dimension is some value $k$, you need to show that there is some set of size $k$ that is shattered, and that no set of size $k + 1$ is shattered.

3. (5 pts) Perceptron algorithm. Implement the Perceptron algorithm presented in class in 2 dimensions and perform the following experiments where concept $C$ is defined by $C(b) = +1$ if $x_1 + x_2 > 0$ and $C(b) = -1$ otherwise.

Experiment 1:

Generate a 50 example training set by picking points uniformly at random from the unit circle and generating labels ($y$-values) according to $C$.

Calculate the gap of the best homogeneous separating line (this gap is the distance between the separating line and the closest example, and the best separating line is not likely to be $C$’s decision boundary).

One way to calculate the gap is the following: for each point with a negative label, multiply its components by -1 and change its class to positive. By doing that we move (about half) of the points of the circle (and get a collection of points that looks like a semicircle.) Next, we find the end points $\bar{x}_l$ and $\bar{x}_r$ of that collection and compute the weight $\bar{w} = \alpha(\bar{x}_l + \bar{x}_r)/2$, which makes the left and right gaps equal. The parameter $\alpha$ is used to normalize the weight. Now the gap may be computed as $\bar{w} \cdot \bar{x}_l / ||\bar{w}||_2$.

Run the Perceptron algorithm and note how many ”mistakes” it makes before finding a consistent hypothesis, and how many iterations through the data are required before it finds a consistent hypotheses.
Perform experiment 1 a hundred times and sort them by the gap. Do you see a relationship between the gaps and the number of iterations of number of mistakes made?

Experiment 2:

Generate a 100 example training set by picking points uniformly at random from the unit circle, with noisy labels. For each example \( b = (x_1, x_2) \) in the training set, generate a random number \( r \) in \([0, 1]\). If \( x_1 + x_2 + 2r - 1 > 0 \) then set the label of \( b \) to 1. If \( x_1 + x_2 + 2r - 1 \leq 0 \) then set the label of \( b \) to -1. Also generate two 100 example test sets, one the same way and one without noise.

This generates a noisy version of \( C \) where the noise tends to be concentrated around the decision boundary.

Run the following version of the perceptron algorithm for 500 iterations where each iteration uses a random point from the training set (rather than cycling through the training set) and save the weight vector \( \bar{w}_i \) after each iteration \( i \). Compare how well the following prediction rules perform on the test sets.

(a) Last hypothesis: predict on \( b \) with \( \text{sign}(\bar{w}_{500} \cdot b) \), using the hypotheses from the last iteration.

(b) Voted hypothesis: predict on \( b \) using the majority of the \( \bar{w}_i \), i.e. \( \text{sign}\left(\sum_{i=1}^{500} \text{sign}(\bar{w}_i \cdot b)\right) \).

(c) Longest survivor: Each \( \bar{w} \) values is created on some iteration \( t \), predicts correctly for a while, and then makes a mistake at some later iteration \( t' \). The survival time of \( \bar{w} \) is \( t' - t \). Let \( \bar{w}_\ell \) be the longest surviving hypothesis from the 500 iterations (pick the first one in case of ties, and assume that the last \( \bar{w} \) makes an incorrect prediction on iteration 501).

(d) (Optional) Suggest your own method and compare its performance with the above.

Create 20 different training and testing sets, and run the perceptron algorithm on each training set. How well do each of the prediction rules do on the test sets?