ECE/CS 559 - Spring 2020 - Midterms # 2 and #3.

Full Name: **ID Number:**

Q1. (34 pts) Let $y = \phi(V\phi(Wx))$, where V and W are matrices of arbitrary dimensions. Write down the Gradient descent update equations for V and W to minimize the function $(V, W) \mapsto \frac{1}{2}\|d-y\|^2$, where d is some fixed vector. Pick $\eta = 1$. For your convenience, I am also defining the function $z = (d-y) \odot \phi'(V\phi(Wx))$. Here, $\odot$ represents the Kronecker product, i.e. for two vectors $x = [x_1 \cdots x_n]^T$, and $[y_1 \cdots y_n]^T$, we define $x \odot y = [x_1 y_1 \cdots x_n y_n]^T$.

Your final expressions should only contain the variables/operators:
$x, y, z, d, V, W, \phi, \phi', (\cdot)^T, \odot, +, -, (,), \leftarrow$.

Do not define intermediate variables/operators of your own!! - otherwise, you will not receive any credit for your answer. You need to write two and only two update equations, one for V and one for W.

Show your work to receive partial credit.

Q2. (33 pts) Consider a single-layer feedforward neural network with n neurons and zero biases. Let $w_1, \ldots, w_n$ denote the neuron weight vectors. Suppose $\|w_1\| = \cdots = \|w_n\|$. Consider a standard competitive learning framework where, for a given input, the winner neuron is the neuron with the largest induced local field. Show that the winner neuron coincides with the neuron that is closest to the input in terms of the Euclidean distance.

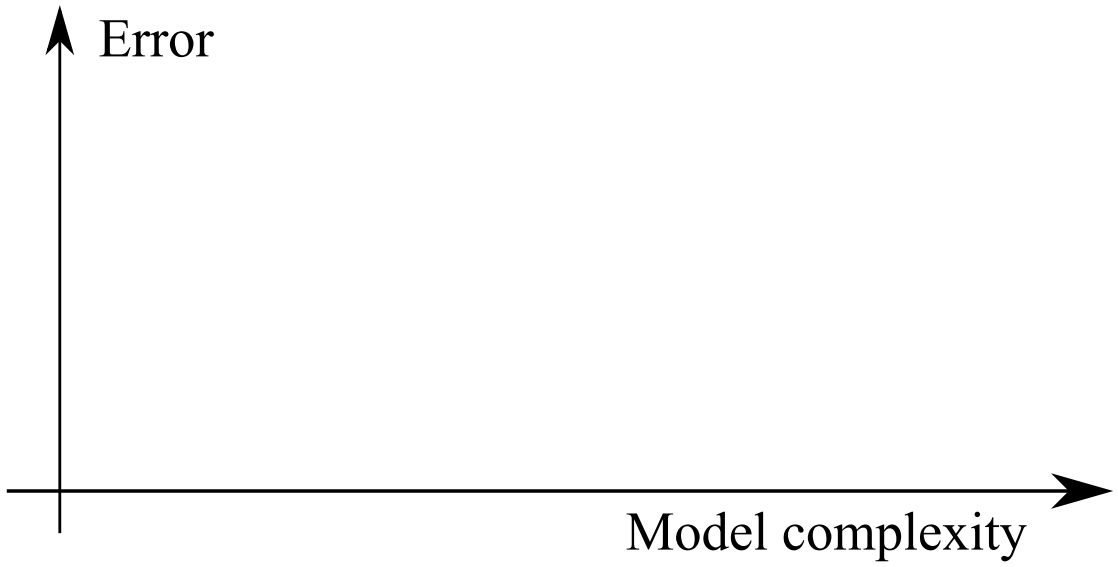
Q3. Consider Hebbian learning for a single neuron with one input, no bias, and one output with step activation function. Suppose that the initial neuron weight w is set to 0.

a) (11 pts) Consider the usual Hebbian learning where the neuron weight is updated as $w \leftarrow w + xy$, where x is the neuron input and y is the neuron output. Suppose that the training sequence is $1, 1, 1, \ldots$ Find the limit of w.

b) (11 pts) Consider the variant where the neuron weight is updated as $w \leftarrow \alpha w + xy$, where $0 < \alpha < 1$. Suppose the training sequence is $1, 1, 1, \ldots$ as before. Find the limit of w.

c) (11 pts) Consider again the variant $w \leftarrow \alpha w + xy$, where $0 < \alpha < 1$. But now, suppose the training sequence is $0, 1, 0, 1, 0, 1, \ldots$. Find the limit of w.

Q4. In this question, we will be drawing curves and labeling regions on the figure below. Suppose we have a labeled dataset, split into a training and testing set, and we aim to learn a classifier on the training set and test its performance on the testing set. Suppose that we can arbitrarily tune the model complexity of our classifier. In the context of neural networks, model complexity may refer to the number of free parameters of the network e.g. the number of neurons, or the number of layers of the network. The more layers or neurons that the network has, the more complex that network is.



(a) (6 pts) Draw a curve on the figure that shows how we would expect the error of the learned classifier on the training data to change as a function of the model complexity. Label this curve in the plot. Briefly describe below your reason for the shape of this curve.

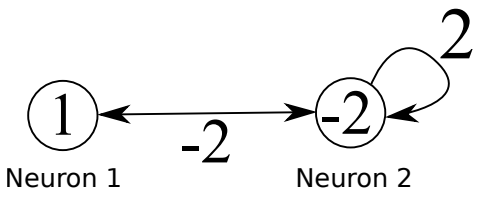
(b) (6 pts) Draw a curve on the figure that shows how we would expect the error of the learned classifier on the testing data to change as a function of the model complexity. Label this curve in the plot. Briefly describe below your reason for the shape of this curve.

(c) (6 pts) Draw a vertical line on the figure that denotes where we would expect to find the optimal model complexity. Label this line in the plot. Briefly describe below your reason for placement of the line.

(d) (6 pts) On the figure, label a region of the x-axis corresponding to underfitting of the training data, and label a region of the x-axis corresponding to overfitting of the training data. Briefly describe below how you chose these regions.

(e) (6 pts) Briefly describe below what would change if we had more training data.

Q5. Consider the Hopfield network below. The activation function is $\phi(x) = 1$ if $x \geq 0$, and $\phi(x) = -1$ if $x < 0$. Recall that the energy function given neuron states $\mathbf{x}$, neuron weight matrix $\mathbf{W}$, and neuron biases $\mathbf{b}$ is given by $E(\mathbf{x}) = -\mathbf{x}^T\mathbf{W}\mathbf{x} - 2\mathbf{b}^T\mathbf{x}$.



(a) (6 pts) Draw the state transition diagram together with state energy levels for the asynchronous update rule. Indicate the steady state(s) of the network.

(b) (6 pts) Does the network always converge to a steady state under the asynchronous update rule? Justify your answer.

(c) (6 pts) Repeat (a) and (b) for the synchronous update rule.

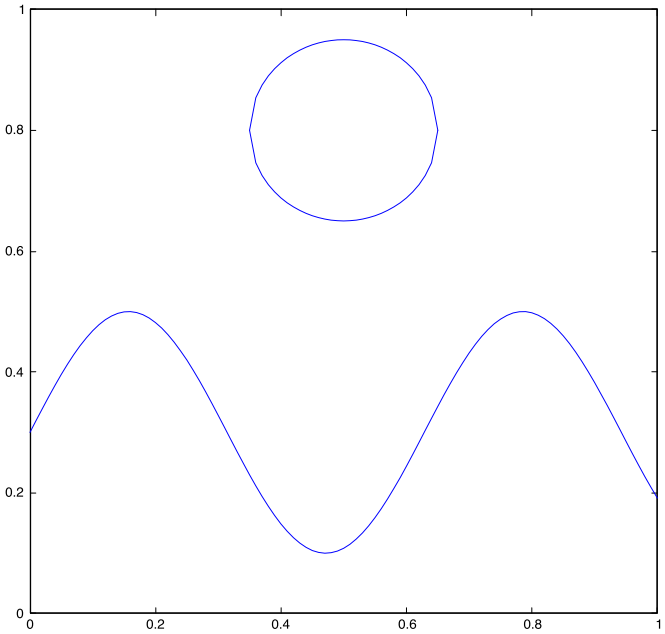
Q6 (52 pts). In this computer project, you will design an SVM. You may use an existing library for solving the quadratic optimization problem that is associated with the SVM. For example, the free software Octave has such a command named "quadprog" to solve such problems. Other than that, you cannot use any existing machine learning/SVM library. As usual, please include the computer codes in your report.

1. Draw 100 points $\mathbf{x}_1, \ldots, \mathbf{x}_{100}$ independently and uniformly at random on $[0,1]^2$. These will be our input patterns.

2. Given $\mathbf{x}_i = \begin{bmatrix} x_{i1} \\ x_{i2} \end{bmatrix}$, let

$$d_i = \begin{cases} 1, & x_{i2} < \frac{1}{5}\sin(10x_{i1}) + 0.3 \text{ or } (x_{i2}-0.8)^2 + (x_{i1}-0.5)^2 < 0.15^2 \\ -1, & \text{otherwise} \end{cases} \tag{1}$$

   denote the desired classes corresponding to the input patterns. Let $C_1 = \{\mathbf{x}_i : d_i = 1\}$ and $C_{-1} = \{\mathbf{x}_i : d_i = -1\}$. The region where d_i is 1 is the union of the region that remains below the mountains and the region that remains inside the sun in the figure below. Sketch the points $\mathbf{x}_i$, indicate (with different colors or markers) their desired class. **Make sure your samples include at least one point inside the sun.**



3. Pick an appropriate kernel (write it down in your report) and design an SVM that separates the classes C_1 and C_{-1}. Recall that the result of the SVM will be a discriminant function $g(\mathbf{x}) = \sum_{i=1}^{\mathcal{I}_s} \alpha_i d_i K(\mathbf{x}_i, \mathbf{x}) + \theta$, where α_i are some positive constants, $\mathcal{I}_s$ is the set of the indices of support vectors, and θ is the optimal bias. Provide a rough sketch of the decision boundaries

$$\mathcal{H} \triangleq \{\mathbf{x} : g(\mathbf{x}) = 0\}$$
$$\mathcal{H}^+ \triangleq \{\mathbf{x} : g(\mathbf{x}) = 1\}$$
$$\mathcal{H}^- \triangleq \{\mathbf{x} : g(\mathbf{x}) = -1\}.$$

together with the support vectors. Note that the support vectors will be either on $\mathcal{H}^+$ or $\mathcal{H}^-$. **Your SVM should be able to perfectly separate the two classes with no exceptions of misclassified input patterns.**

Example solution: What I am expecting is a figure like this. The red crosses are input patterns with desired class 1, and the black diamonds are the input patterns with desired class -1. The decision boundary $\mathcal{H}$ (the blue line) can perfectly separate the two classes that were otherwise not linearly separable. There are 7 support vectors (marked by black circles) on $\mathcal{H}^+$ (the black line), and 6 support vectors (marked by red circles) in $\mathcal{H}^-$ (the red line) for a total of 13 support vectors. As expected, there are no patterns in between the "guard lines" $\mathcal{H}^+$ and $\mathcal{H}^-$. Of course, the Kernel that I used is a secret.

Note that plotting the decision boundary $\mathcal{H} \triangleq \{\mathbf{x} : g(\mathbf{x}) = 0\}$ (or plotting $\mathcal{H}^+$ or $\mathcal{H}^-$) as a nice perfect line is quite a difficult task. What you can do instead is to approximate it as:

for $x_1 = 0, 0.001, 0.002, \ldots, 0.999, 1$
  for $x_2 = 0, 0.001, 0.002, \ldots, 0.999, 1$
    if $g(\mathbf{x})$ is close enough to 0, then put a point in coordinates (x_1, x_2).
  end
end

That is more-or-less what I did in my own figure (That is why the decision boundaries are not perfect and there are some "gaps" in between, you can even see the points that form the lines when you zoom in the PDF). Of course, you can use better techniques for generating the boundary curves if you like.