CMSC 726 : Machine Learning *First Last, UID, email*

HW3

The deadline for submission is 11:59:59 PM, Thursday, October 14.

0.1 Instructions

- Submit your assignment via ELMS, at `http://elms.umd.edu`.
- Your submission should be a single PDF write-up. Since this is a single file, please do *not* zip it.
- Do *not* submit the .tex file.
- You may use LaTex or Word to create your write-up, but **you must submit in PDF format**.
- For your convenience, you can simply edit the included .tex file. To do so,
  1. Edit the header information with your own personal information (otherwise, I won't know whose submission it is).
  2. Uncomment the `\begin{solution} ... \end{solution}` blocks and fill in your solution. (Note: you don't *need* to use the solution blocks if you find the formatting too restrictive. If you find that your solution spans multiple pages, you can break the solution block into multiple parts. Also, feel free to use `\begin{proof} ... \end{proof}` for proofs.)
- To add an image to this tex file, use the command `\includegraphics{filename}` (search the LaTeX documentation for additional arguments).

1 Neural Nets and Backprop

Consider a two-layer feedforward NN with two inputs x1 and x2, and one hidden unit h1. There are five weights, w_0, w_1, w_2, h_0, h_1, each initialized to 0.1. You are given the following training examples:

```
x1  x2  y
1   0   1
0   1   0
```

Exercise 1 Give the values of the weights after each of the first two training iterations (one iteration means processing one example) of backpropagation. Assume a learning rate of $\eta = 0.3$ and a momentum of $\alpha = 0.9$.

2 α-Networks and Memory

Consider the phrase "Yes, we can". Probably for many of you, it immediatelly brings to your thought American elections, politics etc. This is a remarkable fact since by a small phrase we can retrieve many different memories. Another example is the phrase "To be or not to be". It brings to our minds Shakespeare, literature, England etc. In this exercise we will first define a simple model of memory and investigate its properties, step by step.

Our memory model which we are going to call α-network is a weighted, undirected graph $G(V, E, w)$ where $w : E \to \mathbb{R}$. The edges of our graph are allowed to have negative weights. Each node $v \in V$ will have two states. We model this by assuming that each node can have value either 1 or -1. You can think of each node of the graph as a neuron which can be either "on" or "off"[1]. We will call a configuration S of our network an assignment of ± 1 to our nodes. However, we demand that S respects the following rule:

Rule 1: if an edge (u, v) has negative weight, then u and v must have the **same** state, and if an edge (u, v) has positive weight, then u and v must have the **opposite** state.

A natural first question to ask is therefore if given a graph $G(V, E, w)$ we can always find a configuration S such that it respects the above rule.

Exercise 2a Prove by giving a counterexample that the answer to the above question is negative. In other words, give a network for which you can find no S which respects the aforementioned rule.

Now, let's define a different concept which will have the property in constrast to the above rule to be satisfiable for any graph $G(V, E, w)$. Consider a node v with state $s(v)$. v has a set of neighbors $v_1, \ldots, v_k$, where $k \geq 1$ and their corresponding states are $s(v_1), \ldots, s(v_k)$. We will say that **node v is good** if the following condition holds for v:

$$\sum_{i=1}^{k} w((v, v_i)) s(v) s(v_i) \leq 0 \qquad (1)$$

One way to think about Equation 1 is the following: each node (neuron) wonders "what should my state be, +1 or -1 in order to satisfy as much as I can for Rule 1?". We claim the following: Equation 1 provides a rational strategy for each node to decide its state.

Exercise 2b Justify why the above claim is true, by thinking of each node separately. What happens if it does not behave as described above?

We call **a configuration S for which every node is good α-configuration**. Can we always find an α-configuration for a network? And if yes, how? (Note: this a rhetorical question; not an exercise.) Consider the following simple algorithm:

[1]It cannot be in two different states at the same time.

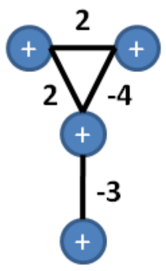

Figure 1: Question **(2)**

```
While the current configuration is not an alpha-configuration
     Choose a node v which is not good
     Flip its state
Endwhile
```

Exercise 2c Run the above algorithm on the network shown in figure 1.

Exercise 2d If you answered Exercise 2c correctly, then you should have seen that the algorithm terminates. It is clear that if the algorithm terminates, it outputs an α-configuration. However it is not clear that the algorithm terminates. Prove that the algorithm terminates.

Hint: Find a quantity which strictly increases at every step of the algorithm and has an absolute upper bound.

3 Support Vector Machines

3.1 Feature Maps and Kernels

For the following questions, you are given a data set D in fig:svmDataset with data from a single feature X_1 in $\mathbb{R}^1$ and corresponding label $Y \in \{+, -\}$. The data set contains four positive examples at $X_1 = \{-3, -2, 3\}$ and three negative examples at $X_1 = \{-1, 0, 1\}$.

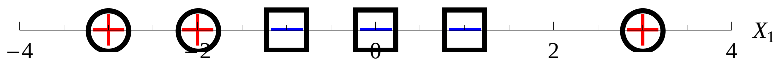

Figure 2: Dataset for SVM feature map task in Section 2.

Exercise 3a Can this data set (in its current feature space) be perfectly separated using a linear separator? Why or why not?

Exercise 3b Lets define the simple feature map $\phi(u) = (u, u^2)$ which transforms points in $\mathbb{R}^1$ to points in $\mathbb{R}^2$. Apply ϕ to the data and plot the points in the new $\mathbb{R}^2$ feature space.

Exercise 3c Can a linear separator perfectly separate the points in the new $\mathbb{R}^2$ features space induced by ϕ? Why or why not?

Exercise 3d Give the analytic form of the kernel that corresponds to the feature map ϕ in terms of only X_1 and X_1'. Specifically define $k(X_1, X_1')$.

Exercise 3e Construct a maximum-margin separating hyperplane. This hyperplane will be a line in $\mathbb{R}^2$, which can be parameterized by its normal equation, i.e. $w_1Y_1 + w_2Y_2 + c = 0$ for appropriate choices of w_1, w_2, c. Here, $(Y_1, Y_2) = \phi(X_1)$ is the result of applying the feature map ϕ to the original feature X_1. Give the values for w_1, w_2, c. Also, explicitly compute the margin for your hyperplane. You do not need to solve a quadratic program to find the maximum margin hyperplane. Note that the line must pass somewhere between (-2,4) and (-1,1) (why?), and that the hyperplane must be perpendicular to the line connecting these two points. Use only two support vectors.

Exercise 3f On the plot of the transformed points (from 3b), plot the separating hyperplane and the margin, and circle the support vectors.

Exercise 3g Draw the decision boundary separating of the separating hyperplane, in the original $\mathbb{R}^1$ feature space.

Exercise 3h Compute the coefficients α and the constant b in Equation 2 for the kernel k and the support vectors $SV = \{u_1, u_2\}$ you chose in exercise 3e. Be sure to explain how you obtained these coefficients.

$$y(x) = \text{sign}\left(\sum_{n=1}^{|SV|} \alpha_n y_n k(x, u_n) + b\right) \tag{2}$$

Think about the dual form of the quadratic program and the constraints placed on the α values.

Exercise 3i If we add another positive $(Y = +)$ point to the training set at $X_1 = 5$ would the hyperplane or margin change? Why or why not?

3.2 Kernel Translation

Three different support vector machines have been trained on a 2D data set using:

1. a linear kernel, $k(\mathbf{x}, \mathbf{y}) = \mathbf{x}^T\mathbf{y}$ (Figure 3a)
2. a quadratic polynomial kernel, $k(\mathbf{x}, \mathbf{y}) = (\mathbf{x}^T\mathbf{y} + 1)^2$ (Figure 3b)
3. an RBF kernel, $k(\mathbf{x}, \mathbf{y}) = exp\big(-\frac{1}{2\sigma}(\mid \mathbf{x} - \mathbf{y} \mid)^2\big)$ (Figure 3c)

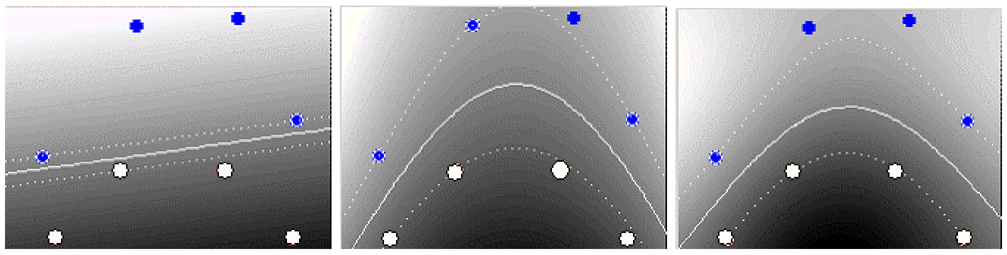

Figure 3: 3 Kernels: (a) Linear, (b) Quadratic and (c) RBF

Assume we now translate the data by adding a large constant value (i.e. 10) to the vertical coordinate of each of the data points, i.e. a point (x,y) becomes (x,y+10).

Exercise 3j If we retrain the above SVMs on this new data, do the resulting SVM boundary change relative to the data points? Say if it does change or not for case (a), case (b) and case (c). Briefly explain why or why not it changes for all 3 cases (a), (b) and (c) and suggest or draw what happens to the resulting new boundaries when appropriate.