ECE/CS 498 DSU/DSG Spring 2020
In-Class Activity 6

NetID:

The purpose of the in-class activity is for you to:
(i) Review concepts related to SVM and neural networks
(ii) Work out steps in backpropagation for optimization of a neural network

Support Vector Machine

1. Linear decision boundary

Sketch the hyperplane $-2 + x_1 + 2x_2 = 0$. Indicate the set of points for which $-2 + x_1 + 2x_2 > 0$, as well as the set of points for which $-2 + x_1 + 2x_2 \leq 0$.

2. Non-linear decision boundary

a) Sketch the curve $(1 + x_1)^2 + (2 - x_2)^2 = 4$.

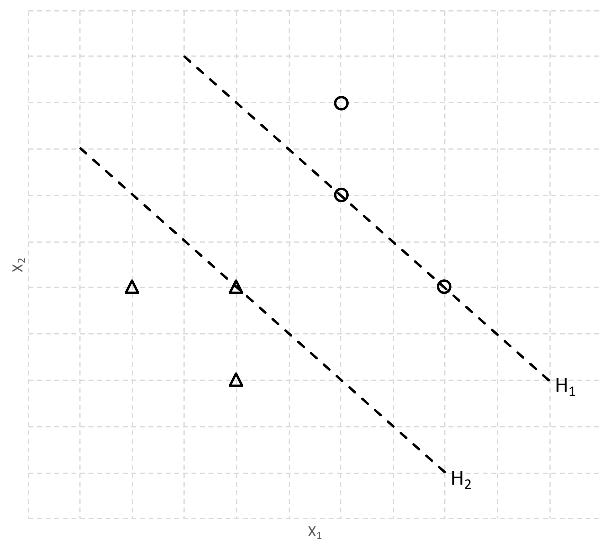
b) On your sketch, indicate the region for which $(1 + x_1)^2 + (2 - x_2)^2 > 4$, as well as the region for which $(1 + x_1)^2 + (2 - x_2)^2 \leq 4$.

c) Suppose that a classifier assigns an observation (x_1, x_2) to the blue class if $(1 + x_1)^2 + (2 - x_2)^2 > 4$, and to the red class otherwise. To what class is the observation $(0, 0)$ classified? $(-1, 1)$?

d) The decision boundary equation in c) is not linear in terms of an input $\boldsymbol{x} = (x_1,\ x_2)$ since it has x_1^2 and x_2^2 terms. However, suppose that we instead consider $\boldsymbol{x} = (x_1,\ x_1^2,\ x_2,\ x_2^2)$. This might be the case after applying a kernel transformation to the dataset. Is the decision boundary equation linear then? What might be a reason for applying this kernel transformation?

3. Hard margin SVM

Suppose we are learning a hard margin SVM with two real-valued features x_1, x_2 and binary label $y \in \{-1, +1\}$ (represented by △ and ○, respectively). The training data is pictured in the figure below. Our linear classifier takes the form $\boldsymbol{w} \cdot \boldsymbol{x} + b = 0$.



a) According to the maximum margin principle, identify the support vectors, and sketch the decision boundary of the trained SVM.

b) Suppose hyperplane H_1 takes the form $\boldsymbol{w} \cdot \boldsymbol{x} + b = 1$, write down the equations for H_2.

c) The constraints for linear hard margin SVM can be written as $y_i(\boldsymbol{w} \cdot \boldsymbol{x_i} + b) \geq 1$, $\forall i \in \{1,\ \ldots,\ N\}$. Explain why.

d) What condition do the data points have to satisfy such that a feasible $\boldsymbol{w}$ exists?

e) Calculate the distance between H_1 and H_2.

f) Based on your answer to the above questions, write down the optimization problem whose solution is hard margin SVM.

g) For the following data points $\boldsymbol{x_1} = (\mathbf{1}, \mathbf{2}, \mathbf{3}), \boldsymbol{x_2} = (\mathbf{4}, \mathbf{1}, \mathbf{2}), \boldsymbol{x_3} = (-\mathbf{1}, \mathbf{2}, -\mathbf{1})$ corresponding to class $y_1 = +1, y_2 = +1, y_3 = -1$, one of the following $\boldsymbol{w}$, b gives the correct SVM decision boundary ($\boldsymbol{w} \cdot \boldsymbol{x} + b = 0$). Which one is it? Show your work.

A. $w = [0.3, 0, 0.4]'$, $b = -0.4$

B. $w = [0.2, 0, 0.4]'$, $b = -0.4$

C. $w = [0.1, 0, 0.4]'$, $b = -0.4$

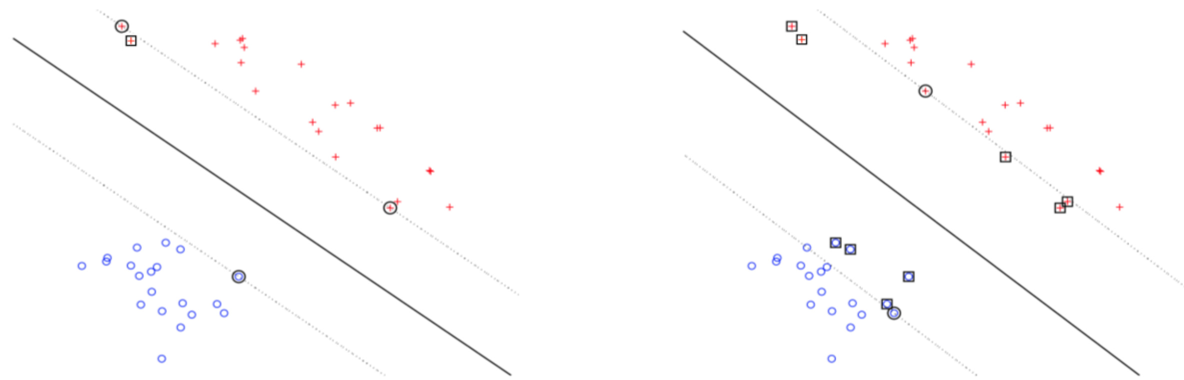
D. $w = [0.4, 0, 0.2]'$, $b = -0.4$

4. Soft margin SVM

Recall the program for solving the soft margin SVM:

$$\min_{\boldsymbol{w},\ \xi_I \geq 0} \frac{1}{2}\left|\left|\boldsymbol{w}\right|\right|^{\mathbf{2}} + C \sum_i^N \xi_i \qquad s.t. \quad 1 - \xi_i \leq y_i(\boldsymbol{w}^T \boldsymbol{x_i} + b) \quad \forall i \in \{1,\ \ldots,\ N\}$$

We have plotted the SVM solutions for a training dataset in Figure (a) and (b) corresponding to two values of C:



(a) (b)

a) Indicate non-zero ξ_is on both plots (a) and (b).

b) Which figure corresponds to a larger value of C? Explain why.

The unconstrained form of the above optimization problem is given as:

$$\min_{\boldsymbol{w},\mathrm{b}} \frac{1}{2}\left|\left|\boldsymbol{w}\right|\right|^{\mathbf{2}} + \mathrm{C}\sum_i^N \max\{0, 1 - y_i(\boldsymbol{w}^T \boldsymbol{x_i} + b)\}$$

c) Draw the function $g(z) = \max\{0, \mathrm{z}\}$ for scalar variable z. What is the derivative $\frac{dg(z)}{dz}$? Draw the derivative.

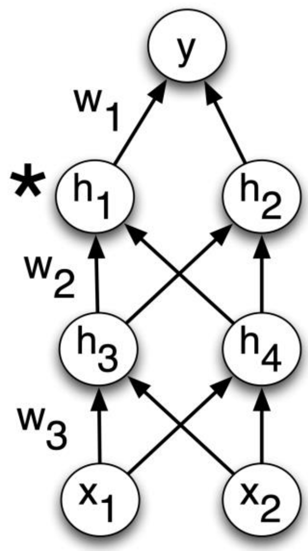
d) Compute the gradient of this unconstrained program w.r.t $\boldsymbol{w}$. [Hint: Think of z in part (c) as $1 - y_i(\boldsymbol{w}^T\boldsymbol{x_i} + b)$, you can express max in terms of an indicator function]

e) Suppose you are training your SVM using gradient descent and the gradient derived in (d), the original gradient is $\boldsymbol{w} = [2,2]', b = -1$. The first iteration we trained on data point $\boldsymbol{x_1} = (1,1), y_1 = 1$; the second iteration we trained on data point $\boldsymbol{x_2} = (-1,-1), y_2 = -1$, $\lambda = 1$, assuming both C and learning-rate to be 1 calculate the gradient after these two iterations.

Neural Networks

1. Partial derivatives

   Consider the network shown in the figure below. All the hidden units use the ReLU- $h_i(\mathrm{z_i}) = \max\{z_i, 0\}$. We are trying to minimize a cost function C which depends only on the activation of the output unit y. The unit h_1 (marked with a *) receives an input of $z_i = -1$ on a training iteration, so its output is 0. Based only on this information, which of the following weight derivatives are guaranteed to be 0 for this training case? Write YES or NO for each. Justify your answers informally. (Hint: don't work through the backprop computations. Instead think about what the partial derivatives really mean.)



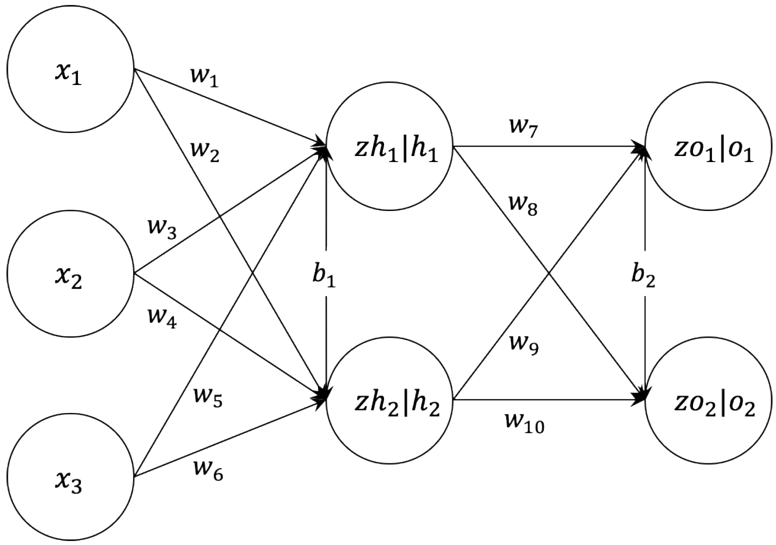
a) $\frac{\partial C}{\partial w_1}$

b) $\frac{\partial C}{\partial w_2}$

c) $\frac{\partial C}{\partial w_3}$

2. Backpropagation

The neural network considered in this question has three input neurons, one hidden layer with two neurons, and one output layer with two neurons. b_1 and b_2 are bias terms.



a) Suppose $zh_1 = w_1x_1 + w_3x_2 + w_5x_3 + b_1$, $h_1 = \text{sigmoid}(\text{zh}_1)$, and similar relationships hold for the other neurons in the hidden/output layer. Assume the current parameters of the networks $w_1, w_2, w_3, w_4, w_5, w_6, w_7, w_8, w_9, w_{10} = 0.1, 0.2, 0.3, 0.4, 0.5, 0.6, 0.7, 0.8, 0.9, 0.1$, $b_1, b_2 = 0.5, 0.5$. Given input $x_1, x_2, x_3 = 1, 4, 5$, use forward propagation to find out the value of h_1, o_1. You can think of writing a Python program.

b) Let t_1, t_2 represent the true labels. Define the sum of squared loss $E = \frac{1}{2}((o_1 - t_1)^2 + (o_2 - t_2)^2)$. **Write down the partial derivatives** $\boldsymbol{\frac{\partial E}{\partial w_7}, \frac{\partial E}{\partial \mathrm{b}_2}, \frac{\partial E}{\partial w_1}}$ **according to the chain rule**. Example for $\frac{\partial E}{\partial w_{10}}, \frac{\partial E}{\partial w_2}$ is given below.

$$\frac{\partial E}{\partial w_{10}} = \frac{\partial E}{\partial o_2} \frac{\partial o_2}{\partial zo_2} \frac{\partial zo_2}{\partial w_{10}}$$

$$\frac{\partial E}{\partial w_2} = \frac{\partial E}{\partial o_2} \frac{\partial o_2}{\partial zo_2} \frac{\partial zo_2}{\partial h_2} \frac{\partial h_2}{\partial zh_2} \frac{\partial zh_2}{\partial w_2} + \frac{\partial E}{\partial o_1} \frac{\partial o_1}{\partial zo_1} \frac{\partial zo_1}{\partial h_2} \frac{\partial h_2}{\partial zh_2} \frac{\partial zh_2}{\partial w_2}$$

For $\frac{\partial E}{\partial w_2}$, look at the two paths which lead from w_2 to E. Backpropagation includes both the paths in the calculation.

c) Suppose $t_1, t_2 = 0.1, 0.05$, learning rate $\alpha = 0.01$. Calculate the updated w_7, b_2, w_1 after one iteration of backpropagation. Use $h_2 = 1$, $o_2 = 0.8$.

3. Neural Network Playground (for you to explore)
https://developers.google.com/machine-learning/crash-course/introduction-to-neural-networks/playground-exercises