DSC 140B - Homework 06

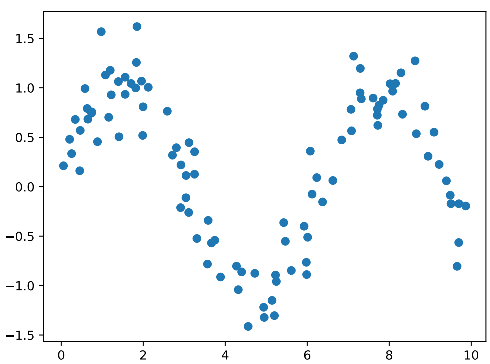
Due: Wednesday, February 18

Instructions:

- Write your solutions to the following problems **by hand**, either on on another piece of paper that you scan or using a tablet. Typed solutions will not be accepted for credit!
  - Code listings are an exception. You do not need to handwrite code, and you can instead include the code as a screenshot.
- Unless otherwise noted by the problem's instructions, show your work or provide some justification for your answer.
- Homework problems are graded pass/fail on completeness and effort, not correctness.
- Homeworks are due via Gradescope at 11:59 PM.

Problem 1. (1 credit)

This problem will use the data shown below:



You can find this data at:

`https://f000.backblazeb2.com/file/jeldridge-data/008-noisy_sin/data.csv`

In each part below, you will train a Gaussian RBF network H on this data using three Gaussians centered at 2, 5, and 8. For each part, plot $H(x)$ for x from 0 to 10, on top of a scatter plot of the data. Your plot should also show the individual Gaussian basis functions, each scaled by its learned weight. That is, your plots should include four functions:

- $w_1 \varphi_1(x)$, where φ_1 is the Gaussian RBF centered at 2 and w_1 is its learned weight;
- $w_2 \varphi_2(x)$, where φ_2 is the Gaussian RBF centered at 5 and w_2 is its learned weight;
- $w_3 \varphi_3(x)$, where φ_3 is the Gaussian RBF centered at 8 and w_3 is its learned weight;
- $H(x) = w_1 \varphi_1(x) + w_2 \varphi_2(x) + w_3 \varphi_3(x)$.

Plot H as a solid line, and the rest as dashed lines. You can use different colors for each function if you like, but it's not required.

a) Train the model using a width parameter of $\sigma = 4$. Show your plot.

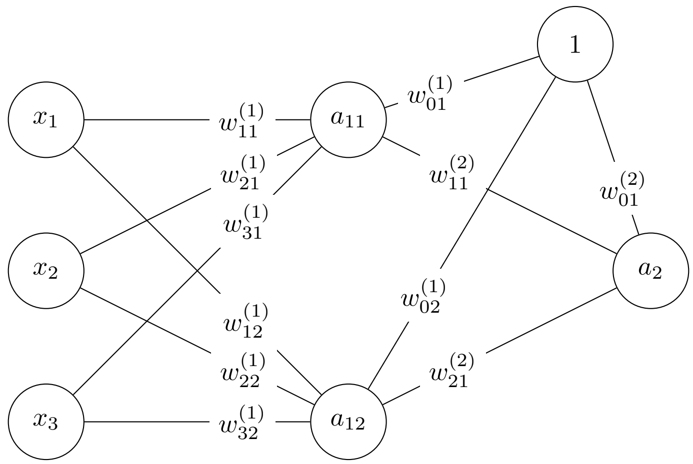
b) Train the model using a width parameter of $\sigma = 1$.

Problem 2. (1 credit)

In lecture it was said that a neural network with linear activation functions is a linear prediction function, meaning that its decision boundary will also be linear. If we wish to have a non-linear decision boundary, we must introduce non-linearities with, for instance, non-linear activation functions.

In this problem, we'll see concretely that a neural network with linear activations is again linear.

Consider the neural network shown below.



The inputs x_1, x_2, and x_3 are numbers. Each $w_{ij}^{(k)}$ is a scalar weight. a_{ij} denotes the output of a neuron. Remember that when linear activations are used, the output of a neuron is simply the weighted sum of its inputs. So for instance:

$$a_{11} = w_{11}^{(1)} x_1 + w_{21}^{(1)} x_2 + w_{31}^{(1)} x_3 + w_{01}^{(1)}$$

The node labeled 1 is the bias input. a_2 is the output of the neural network overall.

a) Write the output of the network, a_2, as an expression involving only the inputs x_1, x_2, x_3 and the weights, $w_{ij}^{(k)}$. a_i should not appear in your expression.

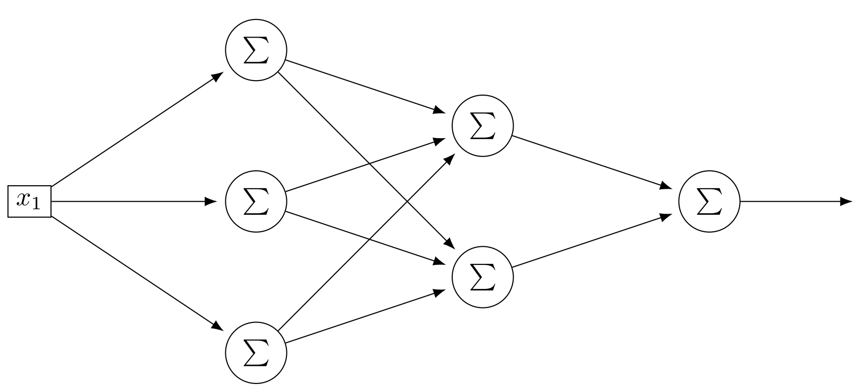
b) Show that the output of the network can be written

$$a_2 = w_0 + w_1 x_1 + w_2 x_2 + w_3 x_3,$$

where w_0, w_1, w_2, and w_3 are scalars that depend only on the weights in the original network. By showing this, you're proving that the network above is equivalent to a much simpler linear model.

Problem 3. (1 credit)

Consider the neural network architecture shown below:



In all parts of this problem, assume that the network's parameters are:

$$W^{(1)} = \begin{pmatrix} -3 & -5 & 4 \end{pmatrix} \qquad W^{(2)} = \begin{pmatrix} 2 & -3 \\ -5 & 5 \\ -3 & 0 \end{pmatrix} \qquad W^{(3)} = \begin{pmatrix} 5 \\ -4 \end{pmatrix}$$

$$\vec{b}^{(1)} = \begin{pmatrix} 3 \\ 2 \\ -1 \end{pmatrix} \qquad \vec{b}^{(2)} = \begin{pmatrix} 1 \\ 2 \end{pmatrix} \qquad \vec{b}^{(3)} = \begin{pmatrix} -3 \end{pmatrix}$$

Note that this network is a function $H : \mathbb{R} \to \mathbb{R}$, so we can easily plot it. In the parts below, you will plot the network for a range of inputs. Your plots may not necessarily be *interesting*, but they will give you a sense of how different choices of activation function affect the type of function the neural network computes.

Suggestion: create a Python function `network(x, activation)` which takes in two things: a number `x` and an `activation` function, and computes the output of the network on x using that activation function (that is, it computes $H(x)$). You can then use that code for all parts of this problem.

a) Assume that all activation functions are linear. Plot $H(x)$ in the range $x \in [-3, 3]$. Show your code.

b) Assume that all hidden nodes have **sigmoid** activation and that the output node has linear activation. Plot $H(x)$ in the range $x \in [-3, 3]$. Show your code.

c) Assume that all hidden nodes have **ReLU** activation and that the output node has linear activation. Plot $H(x)$ in the range $x \in [-3, 3]$. Show your code.