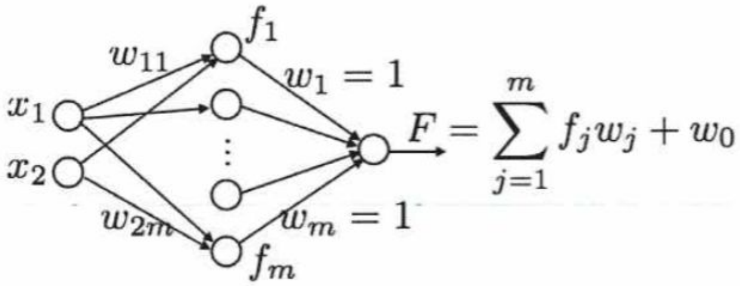
PROBLEM 7

Problem 2 Consider a simple two layer neural network for classifying points on the plane. Our network has additional constraints beyond the two-layer architecture. The main constraint is that all the incoming weights to the output layer, w_j, $j = 1, \ldots, m$, are set equal to one, save for the offset parameter w_0 which remains adjustable. The hidden layer units can be chosen arbitrarily, including their number m.



$$F = \sum_{j=1}^{m} f_j w_j + w_0$$

The weights w_{ij}, $i = 0, 1, 2$, $j = 1, \ldots, m$, can be chosen as needed where w_{0j} is the offset parameter for the j^{th} hidden unit.

(2.1) **(2 points)** If the activation function is sign$(\cdot)$, hidden units act as linear classifiers and can be drawn graphically as such. Write down the normal vector to the decision boundary of the linear classifier corresponding to the i^{th} hidden unit?

$\theta = \begin{bmatrix} W_{1i} \\ W_{2i} \end{bmatrix}$, $\theta_0 = W_{0i}$ $\Rightarrow$ $f_i = \text{sign}(x \cdot \theta + \theta_0)$

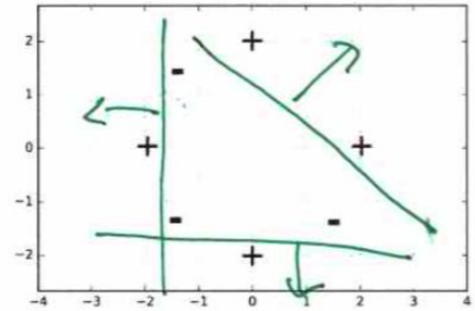
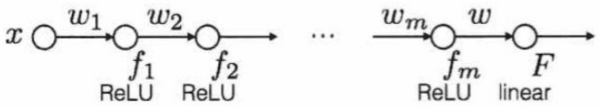


Figure NN1: training set of points

(2.2) **(6 points)** Figure NN1 shows the points we wish to classify correctly. Please draw graphically (as linear classifiers, including orientation) the smallest number of hidden units that enable our constrained output layer to classify the points correctly. For this part you must assume that hidden units are ReLU units.

(2.3) **(4 points)** Neural networks are powerful but can be challenging to train. Consider a simple deep architecture shown below where there is a single unit in each layer.



Each unit uses ReLU activation such that $f_i = \text{ReLU}(w_i f_{i-1} + w_{i0})$, $i = 1, \ldots, m$, where $f_0 = x$. The output unit is linear $F = w f_m + w_0$. For a given input x, we observe target output y, and measure loss $\text{Loss}(F, y)$, where F is the activation of the final linear unit in response to x. In order to train these models with gradient descent, we must be able to calculate gradients. Let

$$d_i = \frac{\partial}{\partial f_i} \text{Loss}(F, y), \quad i = 1, \ldots, m \tag{1}$$

Write down a gradient descent update rule for parameter w_1 using d_i, $i = 1, \ldots, m$.

$$w_1 \leftarrow w_1 - 2 \frac{\partial \text{Loss}(F,y)}{\partial w_1} = w_1 - 2 \frac{\partial f_1}{\partial w_1} d_1 = w_1 - 2 x [\![f_1 \geq 0]\!] d_1$$

(2.4) **(4 points)** Suppose $\frac{\partial}{\partial F} \text{Loss}(F, y) = 1$, i.e., we didn't quite predict the response correctly. In this case, which of the following statements are necessarily true in our deep architecture? Check all that apply.

-2 for any mistake

( ) $d_{i-1} = w_i \, f_i \, d_i, \quad i = 2, \ldots, m$
(✓) $d_{i-1} = w_i \, [\![f_i \geq 0]\!] \, d_i, \quad i = 2, \ldots, m$
(✓) $d_m = w$
( ) $d_1 \rightarrow 0$ as m increases (vanishing gradient)
(2)

7)

a) Recall that the normal vector to a hyperplane is defined as the θ parameter. The hidden sign units in this network perform the same kind linear combination with an offset that our signed hyperplanes use.

b) Using the fact that the hidden units act as linear classifiers, we can draw decision boundaries that separate the signed points (orientation is important). Multiple solutions.

c) Solutions are fine.

d) Solutions are fine.