Homework 3

1 Exercises

1.2.14 For each of the Boolean functions given below, state whether the problem is linearly separable.

a. A AND B AND C

Yes

b. NOT A AND B

Yes

c. (A OR B) AND (A OR C)

Yes

d. (A XOR B) AND (A OR B)

No

1.2.15.b Comment on the disadvantage of using linear functions as activation functions for multi-layer neural networks.

Multilayer neural networks are useful for modeling nonlinear relationships between the input and output attributes. However, if linear functions are used as activation functions (instead of sigmoid or hyperbolic tangent function), the output is still a linear combination of its input attributes. Such a network is just as expressive as a perceptron.

1.3 Consider a dataset that has 8 predictors. You train a neural network with 3 hidden layers and an output layer that predicts a continuous value (a regression problem). The first hidden layer has 16 neurons, the second has 8 neurons, and the third has 4 neurons. In this network, how many total parameters will you have?

First hidden layer (16 neurons):

Weights: There are 8 input features and 16 neurons, so there are $8 \times 16 = 128$ weights.
Biases: There is one bias for each of the 16 neurons, so there are 16 biases.
Total parameters: $128 + 16 = 144$

Second hidden layer (8 neurons):

Weights: There are 16 neurons from the previous layer and 8 neurons in this layer, so there are $16 \times 8 = 128$.
Biases: There is one bias for each neuron, so there are 8 biases.
Total parameters: $128 + 8 = 136$

Third hidden layer (4 neurons):

Weights: There are 8 from the previous layer and 4 in this layer, so there are $8 \times 4 = 32$ weights.
Biases: There is one bias for each neuron, so there are 4 biases.

Total parameters: $32 + 4 = 36$

Output layer (1 neuron):

Weights: There are 4 from the previous layer and 1 from the output layer, so there are $4 \times 1 = 4$ weights.

Biases: There is one bias for the output neuron, so there is 1 bias.

Total parameters: $4 + 1 = 5$

Total number of parameters:

$144 + 136 + 36 + 5 = 321$

2 Programming Problems

Q2(b)(iv) Examine the values for loss and accuracy you got above. Given the shallow network you are using, there are four possible outcomes:

1. High value for accuracy, high value for loss.
2. High value for accuracy, low value for loss.
3. Low value for accuracy, high value for loss.
4. Low value for accuracy, low value for loss.

Where in the above spectrum does your loss and accuracy fall? Do the loss and accuracy meet your expectation?

My loss and accuracy fall into one of the four outcomes listed. The loss and accuracy fall into the 4th outcome: low accuracy (0.25) and high loss (1.56). This suggests that the shallow neural network with only one hidden neuron is underfitting the data, failing to capture the complexity of the classification task effectively.

Q2(b)(v) Examine the distribution of the predicted labels. What pattern do you see in the predicted labels?

The distribution of the predicted labels reveals class imbalance, where the model favors certain room predictions over others. This could be due to the model's inability to learn complex features with just one hidden neuron, leading to biased predictions toward more frequent classes.

Q2(b)(vi) Do you think we will get better results if we increase the training to 200 epochs? Why?

Increasing the training epochs to 200 may improve the results slightly, but the improvement will be marginal. Since the model is very simple, additional epochs could also lead to overfitting, where the model performs well on the training set but does not generalize effectively to the test data.

Q2(c)(iv) Compare the loss and accuracy of your new model with the loss and accuracy obtained from the model in 2(b). Is it better or worse?

The new model had a loss of 0.08 and accuracy of 0.98. This is significantly better than the model from 2(b), which had a loss of 1.56 and accuracy of 0.25. This improvement is due to the addition of a hidden layer with more neurons, which allowed the model to learn more complex patterns in the data.

Q2(c)(vi) Study the confusion matrix you created in 2(c)(ii) and compare it with the confusion matrix you created for the decision tree model in 2(a). Based on the confusion matrices and other knowledge you have gained in class on neural networks, if you had to deploy the model in 2(c) or the model in 2(a) for production, which one would you choose and why?

Based on the confusion matrices, the neural network model (2c) performs better overall compared to the decision tree model (2a). The neural network has higher accuracy in all classes, especially in the class 4, where it achieved a perfect prediction, while the decision tree model had some misclassifications. Given these results, I would choose the neural network model for production because it demonstrates more accurate and consistent predictions across all classes.

CS422 Exercises

1.1 a) $z_1 = w_1 \times x = 1 \times 4 = 4$

$z_2 = w_2 \times x = 1 \times 4 = 4$

$z_3 = w_3 \times x = -1 \times 4 = -4$

From ReLU activation function: $z_1 = 4$, $z_2 = 4$ and $z_3 = 0$

$\rightarrow (z_1 \times w_4) + (z_2 \times w_5) + (z_6 \times w_6)$

$= (4 \times 0.5) + (4 \times 1) + (0 \times 2) = 6$

$\rightarrow \sigma(z) = \frac{1}{1+e^{-z}} = \frac{1}{1+e^{-6}}$

$= 0.998$

b) Loss $= E(y - \hat{y})^2$

$= (y - \sigma)^2$

$= (0 - 0.9975)^2$

$E = 0.996$

c) $\frac{\partial E}{\partial w} = \frac{\partial E}{\partial \sigma(o)} \times \frac{\partial \sigma(o)}{\partial o} \times \frac{\partial o}{\partial w}$

$w_4 = \frac{\partial E}{\partial w_4} = \frac{\partial E}{\partial \sigma(o)} \times \frac{\partial \sigma(o)}{\partial o} \times \frac{\partial o}{\partial w_4}$

$= (-2)(0 - 0.998) \times (0.998)(1 - 0.998) \times 4$

$= 0.016$

$$W_5 = \frac{\partial E}{\partial w_5} = \frac{\partial E}{\partial \sigma(o)} \times \frac{\partial \sigma(o)}{\partial o} \times \frac{\partial o}{\partial w_5}$$

$$= (-2)(0-0.998) \times (0.998)(1-0.998) \times 4$$

$$= 0.016$$

$$W_6 = (-2)(0.998) \times (0.998)(1-0.998) \times 0 \quad = 0$$

$$W_1 = \frac{\partial E}{\partial W_1} = \frac{\partial E}{\partial f(z_1)} \times \frac{\partial f(z_1)}{\partial z_1} \times \frac{\partial z_1}{\partial w_1}$$

$$= \frac{\partial E}{\partial f(z_1)} = \frac{\partial E}{\partial \sigma(o)} \times \frac{\partial \sigma(o)}{\partial o} \times \frac{\partial o}{\partial f(z_1)}$$

$$\frac{\partial E}{\partial f(z_1)} = (-2)(0-0.998) \times (0.998)(1-0.998) \times 0.5 = 0.002$$

$$\frac{\partial E}{\partial w_1} = 0.02 \times 1 \times 4 = 0.08$$

$$W_2 = (-2)(0-0.998) \times (0.998)(1-0.998) \times 1 = 0.04$$

$$\frac{\partial E}{\partial W_2} = 0.04 \times 1 \times 4 = 0.016$$

$$W_3 = (-2)(0-0.998) \times (0.998)(1-0.998) \times 2 = 0.008$$

$$\frac{\partial E}{\partial W_3} = 0.008 \times 0 \times 4 = 0$$

Gradient descent vector: $[0.008, 0.016, 0, 0.016, 0.016, 0]$

d) New weight = Oldweight - learning rate × ∇E

$$W'^T = W^T - \eta \nabla E$$

$$= [1, 1, -1, 0.5, 1, 2] - 1[0.008, 0.016, 0, 0.016, 0.016, 0]$$

$$W'^T = [0.992, 0.984, -1, 0.484, 0.984, 2]$$

Hidden Layer:

$$z_1 = 0.992 \times 4 = 3.968$$

$$\max(0, 3.968) = 3.968$$

$$z_2 = 0.984 \times 4 = 3.936$$

$$z_3 = -1 \times 4 = -4$$

$$\max(0, -4) = 0$$

Final output of hidden layer: $[3.968, 3.936, 0]$

$$z = (0.484 \times 3.968) + (0.984 \times 3.936) + (2 \times 0)$$

$$= 5.793$$

$$\hat{y} = \frac{1}{1+e^{-z}} = \frac{1}{1+e^{-5.793}} = 0.9969$$

Loss error $= (y - \hat{y})^2$

$$= (0 - 0.9969)^2 = 0.994$$

e) Loss value in (b) = 0.996
Loss value in (d) = 0.994

The weight update has reduced the error slightly as expected. But the change is very minimal indicating that we may need to make some changes to the learning rate.