CP8318 Assignment 3
Student Name: Albina Cako

Question 1:
Part 1.

In this part, the graph of loss vs. epoch for the entire training set is shown on the top graph obtained by the unregularized model. Below, the graph of accuracy vs. epoch for the dev set is shown, which was obtained by the unregularized model.

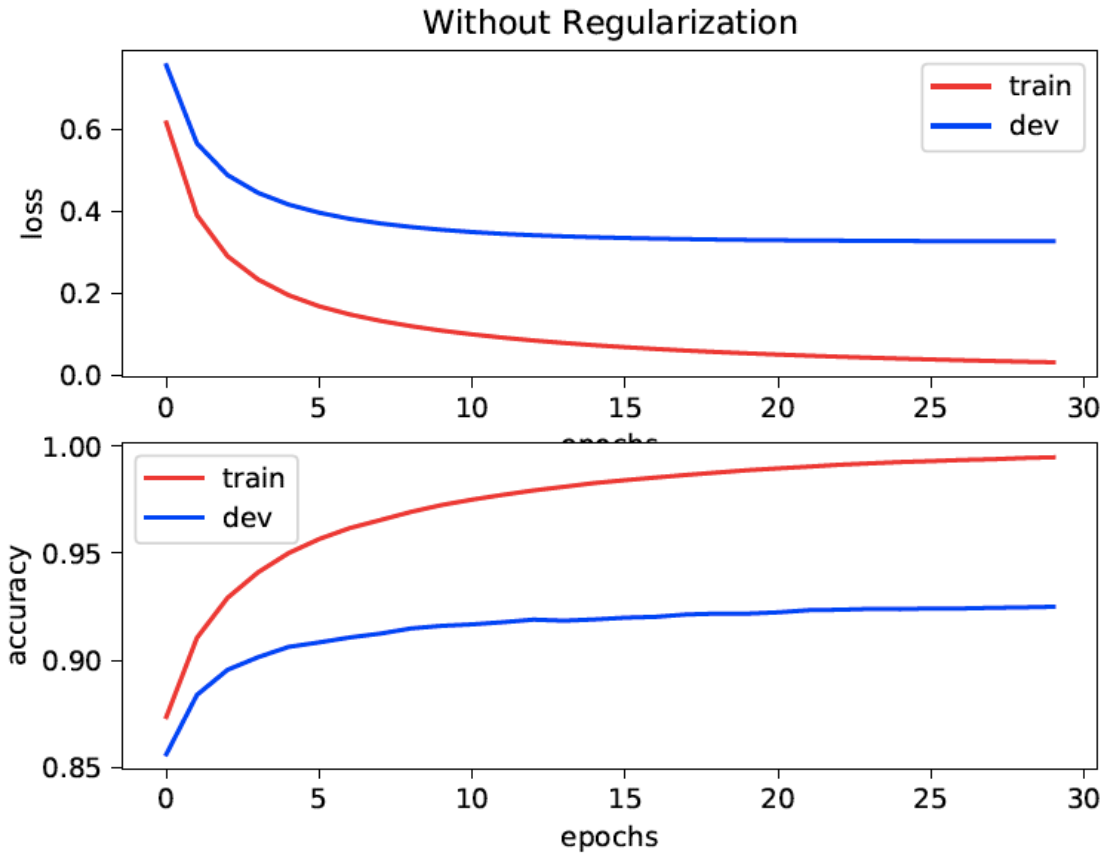


Figure 1. Loss vs. epochs and accuracy vs. epochs of the training set and dev set, respectively. These were obtained by the unregularized model.

Part 2.

In this part, the graph of loss vs. epoch for the entire training set is shown on the top graph obtained by the regularized model. Below, the graph of accuracy vs. epoch for the dev set is shown, which was obtained by the regularized model.

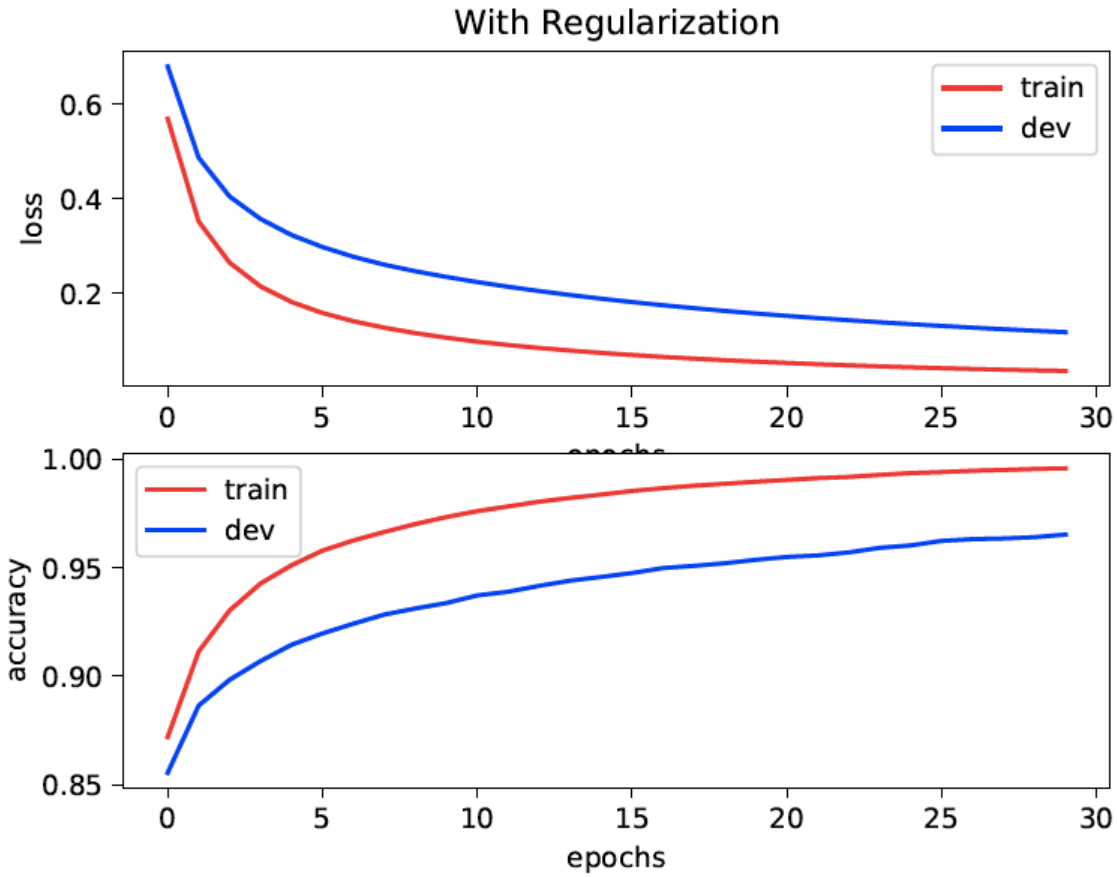


Figure 2. Loss vs. epochs and accuracy vs. epochs of the training set and dev set, respectively. These were obtained by the unregularized model.

By looking at unregularized model on Figure 1, we see that training set loss values are high indicating high bias. The loss value decreases over time as number of epochs increase, showing that the bias gets lower. However, in the dev set, there is high bias at the start and as the number of epochs increases, the los value decreases only slightly over time, which shows high bias remains. In addition, the lines between the training and dev sets in the unregularized model go further apart with time as the number of epochs increases, showing high variance.

On the other hand, on Figure 2, we see that the dev set has a lower loss value over time, showing lower bias then in the unregularized model. In addition, as the number of epochs increases, the distance between the two lines (train vs. dev) is not very different, indicating low variance. Therefore, there is lower variance and lower bias in the regularized model.

To expand further, in the regularized model (Figure 2), the accuracy gap between the training and dev sets is smaller than in the unregularized model (Figure 2). This is due to the fact that regularization helps to avoid model over-fitting.

Part 3.

The test accuracy for the unregularized version of the model is 0.928700.
The test accuracy for the regularized version of the model is 0.967600.

For the unregularized version I got 0.928700, which is similar to 0.9318, but not exact value. This could be due to the fact that we initialized the models parameters at random.

The test accuracy for the regularized model was higher than that of unregularized model as expected. This could due to the fact that using regularization makes the weights very small (however, not 0). If there are outliers, the regularization tries to fix it by penalizing the weights, thus balancing the error differences in the outliers. This could result in the higher accuracy obtained.

Question 2.
Part 1.

The update rule for gradient descent is:

$$\theta_j := \theta_j - \alpha \frac{\partial L}{\partial \theta_j}$$

In this question:
θ_j is $w_{1,2}^{[1]}$
L is l (loss function given)

Let the sigmoid function be:

$$\sigma(x) = \frac{1}{1+e^{-x}}$$

Therefore, we need to find l where $\sigma(x)$ is the activation function.

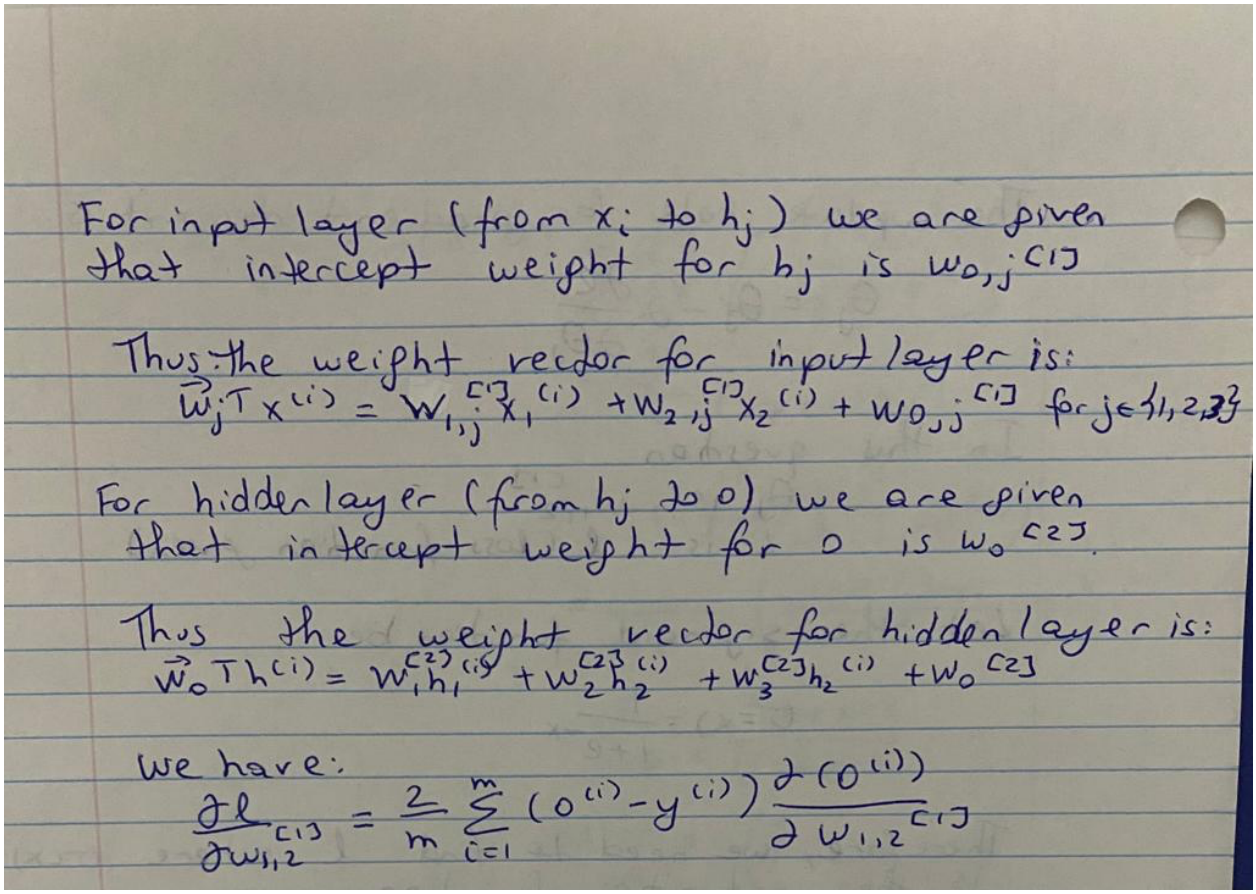
Given: $l = \frac{1}{m} \sum_{i=1}^{m} (o^{(i)} - y^{(i)})^2$

Using chain rule:

$$\frac{\partial l}{\partial w_{1,2}^{[1]}} = \frac{2}{m} \sum_{i=1}^{m} (o^{(i)} - y^{(i)}) \frac{\partial (o^{(i)})}{\partial w_{1,2}^{[1]}}$$

We have 2 layers.
Let $h_j^{(i)}$ represent the hidden neuron j of each sample i.

Given that we have 2 layers, we have to represent each weight as a vector.



Continued on next page:

$\sigma(x)$ is the activation function then

$$o = \sigma(x) = \frac{1}{1+e^{-x}}$$

In order to find $\frac{\partial (o^{(i)})}{\partial w_{1,2}^{[ij]}}$ we first find

$$\frac{\partial o}{\partial x} = \partial(\sigma(x)) = \frac{\partial}{\partial x}\left[\frac{1}{1+e^{-x}}\right]$$

$$= \frac{\partial}{\partial x}(1+e^{-x})^{-1}$$

$$= -1(1+e^{-x})^{-2}(-e^{-x})$$

$$= \frac{1}{1+e^{-x}}\left(\frac{(1+e^{-x})-1}{1+e^{-x}}\right)$$

$$= \frac{1}{1+e^{-x}}\left(\frac{1+e^{-x}}{1+e^{-x}} - \frac{1}{1+e^{-x}}\right)$$

$$= \frac{1}{1+e^{-x}}\left(1 - \frac{1}{1+e^{-x}}\right)$$

Since $\sigma(x) = \frac{1}{1+e^{-x}}$:

$$\frac{\partial o}{\partial x} = \sigma(x)(1-\sigma(x))$$

In hidden layer we know that $x = \vec{w}_o{}^T h^{(i)}$:

$$\frac{\partial o^{(i)}}{\partial w_{1,2}^{[ij]}} = \sigma(\vec{w}_o{}^T h^{(i)})(1-\sigma(\vec{w}_o{}^T h^{(i)}))\, w_2^{[2j]} \frac{\partial (h_2^{(i)})}{\partial w_{1,2}^{[ij]}}$$

Hilroy

In a similar way we obtain $\frac{\partial (h_2^{(i)})}{\partial w_{1,2}^{[1]}}$

With $X = w_2^T x^{(i)}$:

$$\frac{\partial (h_2^{(i)})}{\partial w_{1,2}^{[1]}} = \sigma(\vec{w}_2^T x^{(i)})(1 - \sigma(\vec{w}_2^T x^{(i)}))\, x_1^{(i)}$$

Therefore:

$$\frac{\partial l}{\partial w_{1,2}^{[1]}} = \frac{2}{m}\sum_{i=1}^{m} (o^{(i)} - y^{(i)})\, \sigma(\vec{w}_0^T h^{(i)})(1 - \sigma(\vec{w}_0^T h^{(i)})) \cdot w_2^{[2]} \cdot \sigma(\vec{w}_2^T x^{(i)})(1 - \sigma(\vec{w}_2^T x^{(i)}))\, x_1^{(i)}$$

Therefore the update rule:

$$w_{1,2}^{[1]} := w_{1,2}^{[1]} - \alpha \frac{\partial l}{\partial w_{1,2}^{[1]}}$$

$$w_{1,2}^{[1]} := w_{1,2}^{[1]} - \alpha \left(\frac{2}{m} \sum_{i=1}^{m} (o^{(i)} - y^{(i)}) \sigma(\vec{w}_0^T h^{(i)}) \cdot (1 - \sigma(\vec{w}_0^T h^{(i)})) w_2^{[2]} \cdot \sigma(\vec{w}_2^T x^{(i)})(1 - \sigma(w_2^T x^{(i)}))\, x_1^{(i)} \right)$$

(alpha)

Part 2.

No, it is not possible to have a set of weights that allow the neural network to classify this dataset with 100% accuracy. By looking at the figure in the question, we can see that the classes are not linearly separable. Thus, we cannot use an activation function that is linear, since the classification problem is nonlinearly separable.