Student Name: Yifang Pan
Student Number: 1004192759

Response

3. Solve by Inspection:
In here, we let the output value of $z > 0$ indicate that it is an x, and an output value of $z <= 0$ to indicate that it is not x.

i. Is the answer that we discussed in class unique? If your answer is yes, say why. If not, give a second answer that uses different weights and bias.

The answer we discussed in class is not unique given the output I defined. We can also have [0.95, -1, 1, - 1, 1, -1, 1, -1, 1], and -4 as bias.

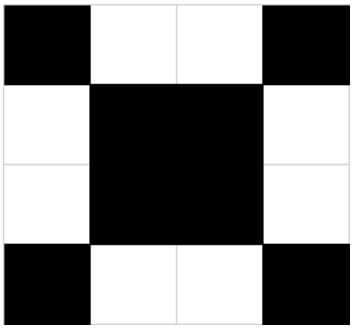
ii. How many unique inputs (that is, different instances of I = {I0,I1,...,I8}) are possible for the 3x3 grid?

There is a total of 2^9 which is 512 inputs for a 3x3 grid.

iii. Does your solution easily scale to solve a 4x4 problem, and an NxN problem? Explain why or why not.

Assuming we are talking about a 4x4 "X" in this case. The 4x4 problem can be solve with a neuron with weight matrixes as shown in figure 1, with black squares having +1 and white squares having -1, while the bias being -7. The NxN problem can be solved similarly, with a bigger "X", and bias equal to number (# of weights equal to 1) – 1.

Figure 1: weight matrix 'X' for 4x4 matrix.



iv. Suppose that, on a 5x5 grid, you had to match the 'X' as above, but, in addition, an 'X' shifted left by one, and shifted right by 1 position also had to be matched. Could you as easily create the single-neuron parameters to solve that problem? Why or why not?

I don't think I would be able to easily create it, because there are 25 parameters to adjust and too many cases to consider. Even if I were to do it. I won't be willing to do it.

6. Experiments and Outputs

Effect of Hyperparameter: Number of Epochs

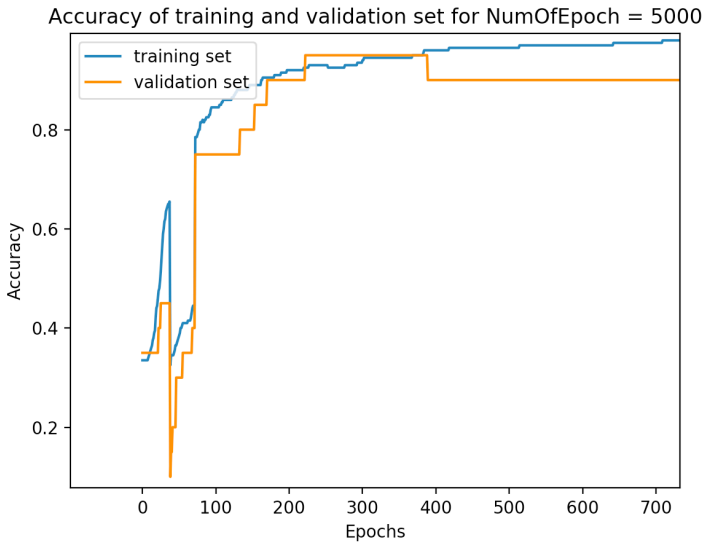
To study the effect of the number of Epochs, I will use the Relu function, a random seed of 0, and a learning rate of 0.001, as a lower learning rate can elongate the learning process and made the trend more pronounced.

Table 1: Effect of different number of epochs on the accuracy of Single Neuron Classifier on training data and validation data

Number of Epochs	10.00	100.00	1000.00	5000.00	10000.00
Accuracy on Training Data	0.34	0.85	0.97	0.98	0.98
Accuracy on Validation Data	0.35	0.75	0.95	0.90	0.90

From the tests done, we can see that at a lower number of epochs, the accuracy is low, this can be due to the fact that we haven't given the classifier enough time to adjust the weights appropriately. In my case, it seems 10 and 100 epochs are not enough to bring the accuracy up to above 90%. Another observation is that the accuracy on the validation data went down after between 1000 epochs and 5000 epochs, which the accuracy on the training data went up. This can be seen as an effect of over-fitting, which is the phenomenon when the classifier loses the ability to generalize, this effect can be seen in figure 2.

Figure 2: Accuracy of training set and validation set to show the reduction of validation set accuracy.



Effect of Hyperparameter: Learning Rate

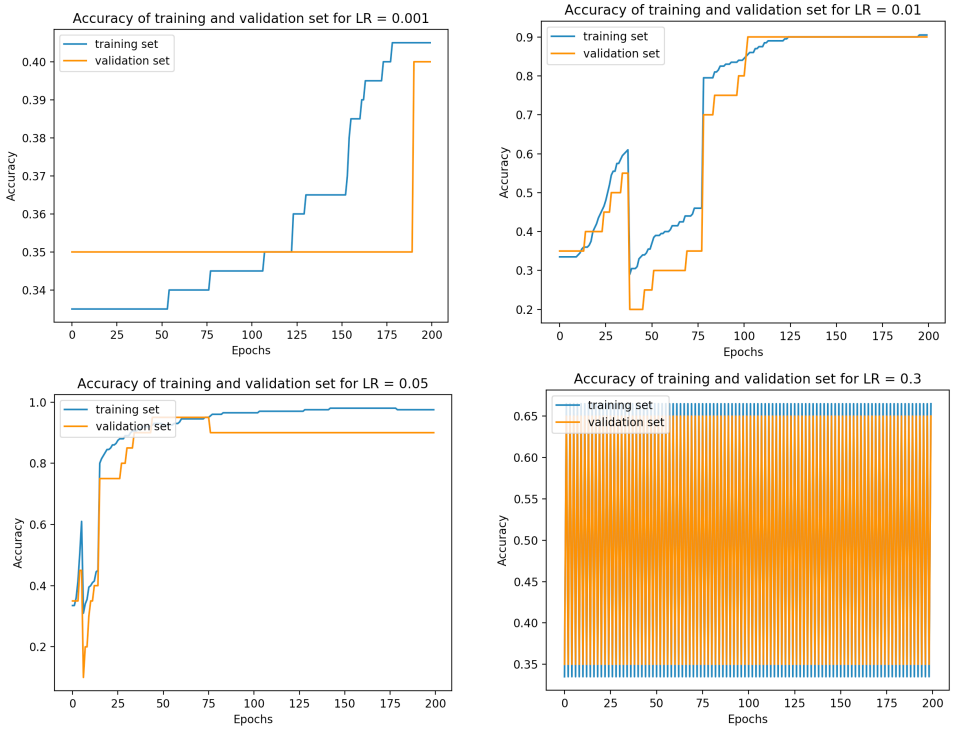
To study the effect of learning rate on classifer, I've set the number of Epochs at 200, the random seed as 0, and using the Relu for the activation function. The resultant data are shown in the table below.

Table 2: Effect of different learning rate on the accuracy of Single Neuron Classifier on training data and validation data

Learning Rate	0.001	0.01	0.01	0.05	0.10	0.20	0.50	1.00
Accuracy on Training Data	0.41	0.90	0.95	0.97	0.98	0.98	0.34	0.34
Accuracy on Validation Data	0.40	0.80	0.90	0.90	0.90	0.90	0.35	0.35

From the table, we can see that the when the learning rate is too low and too high, the classifier is unable to achieve a high accuracy.

Figure 3: Accuracy of training set and validation set for different learning rates



Upon further inspection of the trend in accuracy in training and validation dataset. We can see that for cases where the learning rate is low (not oscillating), the accuracy typically has the same trend, with the difference being how compressed the trend is. From figure 2, we can see the trend with 0.05 as the learning rate is a lot of compressed than the set with 0.01 as the learning rate, which indicates that the learning rate of 0.05 causes the classifier to learn at a faster rate.

The graph gives an intuitive explanation to why the low learning rate classifier yields a low accuracy. From the top left graph, we can see that though the trend is still increasing, it has not yet reached the plateau. This indicates that the trend will still increase given more epochs. When

the learning rate is too high, we can see that the accuracy enters an oscillating pattern. This can be an error during the gradient descent: though the classifier is able to calculate the direction of the gradient correctly, when the classifier adjusts the weights and biases, it will always overshoot as the learning rate is too high. Therefore, the classifier will always oscillate around the local maximum and cause the accuracy to oscillate.

Effect of Hyperparameter: Activation function

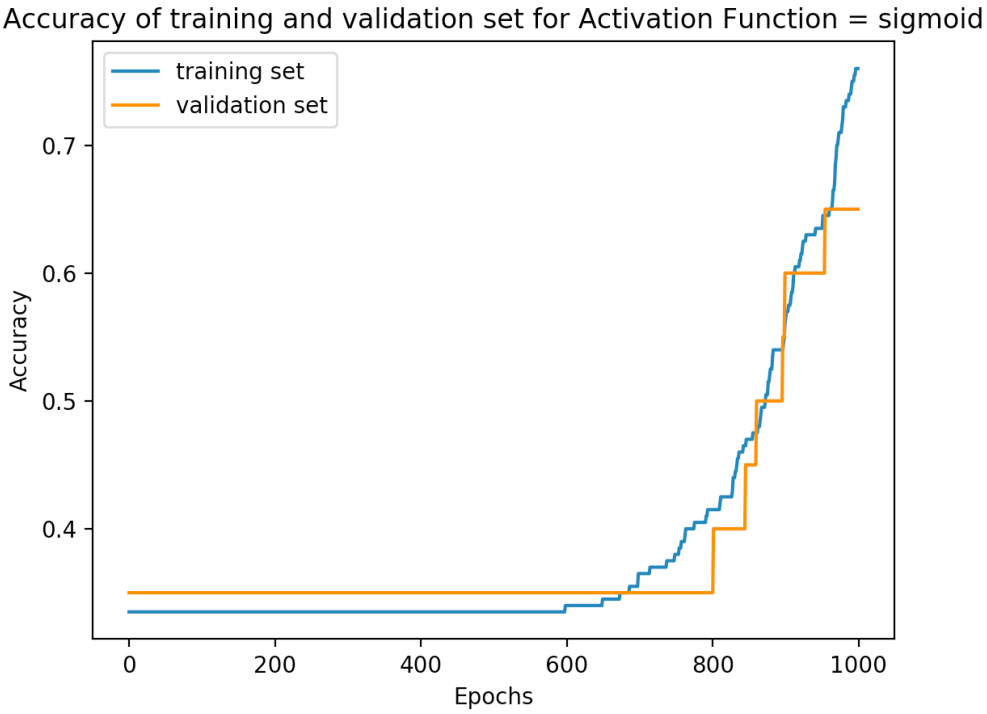
To test out the effect of the activation function, we are using 0.01 for the learning rate, and 1000 as the number of epochs.

Table 3: Effect of different activation functions on the accuracy of Single Neuron Classifier on training data and validation data

Activation Function	Relu	Sigmoid	Linear
Accuracy on Training Data	0.98	0.78	0.97
Accuracy on Validation Data	0.95	0.75	0.95

From the table we can see that the Relu and Linear function has both achieved an accuracy above 90% while the sigmoid function has not. Further inspection of the trend (figure 4) in accuracy of the sigmoid function shows that the sigmoid function has not reached its plateau. This indicates that though the sigmoid function causes a slow gradient descent, given more epochs, the sigmoid function might eventually reach a high accuracy.

Figure 3: Accuracy of training set and validation set for the sigmoid function.



Further testing with 3000 epochs, 0.01 learning rate, and seed of 0 yielded the following data (table 4), confirming that sigmoid function can yield a high accuracy with a higher number of epochs. The reason the sigmoid function being slow might be the fact that the sigmoid function is restricted to between 0 and 1, while the linear and relu function don't have upper limits. For

this reason, when the magnitude of the gradient is very large, the linear and relu function can take advantage of that and speed up the gradient descent and sigmoid cannot.

Table 4: Effect of different activation functions on the accuracy of Single Neuron Classifier on training data and validation data, with 3000 epochs

Activation Function	Relu	Sigmoid	Linear
Accuracy on Training Data	0.99	0.97	0.98
Accuracy on Validation Data	1.00	1.00	0.90

Effect of Random Seeds

To test out the effect of random seeds, I will use the linear function, with a learning rate of 0.01 and 1000 epochs.

Table 5: Effect of different random seed on the accuracy of Single Neuron Classifier on training data and validation data, with 3000 epochs

Random Seed	0	1	2	3	4
Accuracy on Training Data	0.97	0.96	0.98	0.98	0.97
Accuracy on Validation Data	0.90	0.95	0.90	0.95	0.95

The resultant accuracies are either 90% or 95%. The difference can be attributed to the fact the gradient descent algorithm is only good at finding the local minimum as supposed to the global minimum. Therefore, for each trial, the final weight matrix depends on the initial weights, which depends on the randomized seeds.

Best set of Parameters:

The best set of Parameter I found is to use the **Relu** as the activation function, **50** as the number of epochs and **0.08** as the learning rate, with 0 as the random seed. This set of parameter yields an accuracy of **1** for the validation data, and 0.97 for the training data.

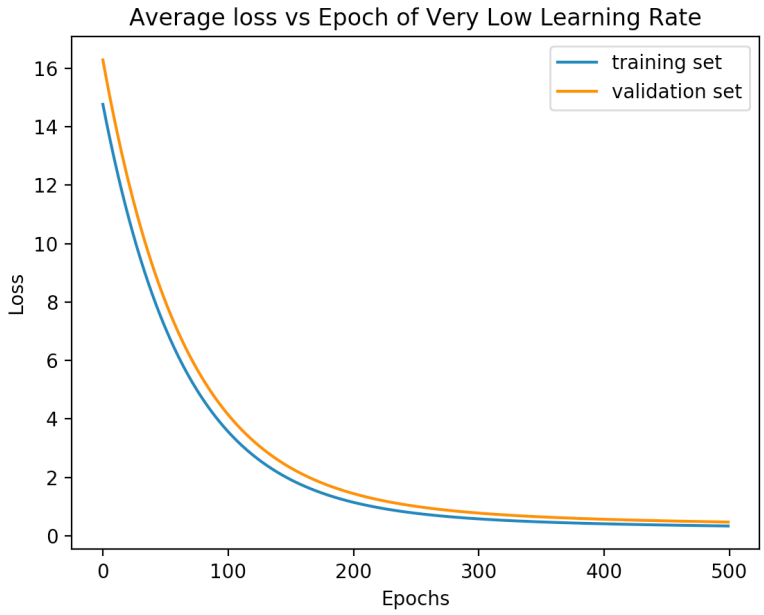
Table 6: Validation accuracy for different number of epochs and different learning rate for relu function

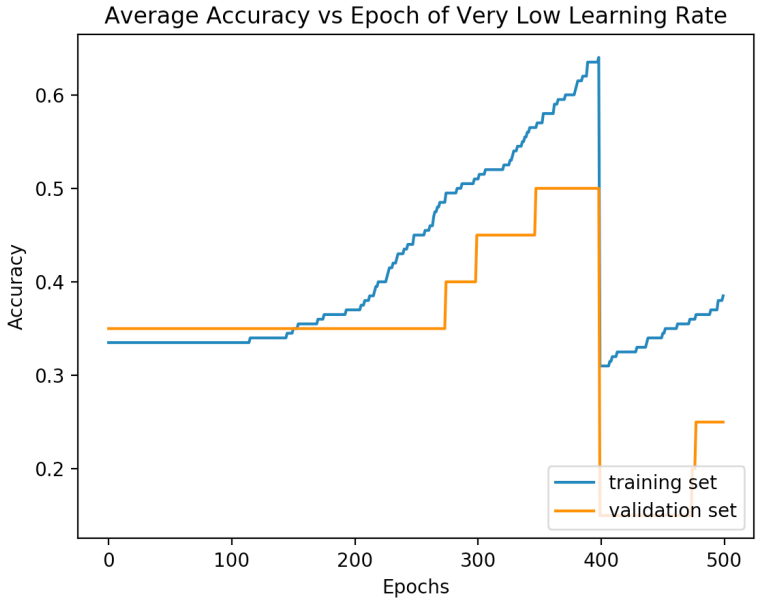
	10	20	30	40	50	60	70	80	90	100	110	120	130	140
0.05	0.35	0.90	0.85	0.90	0.95	0.95	0.95	0.95	0.95	0.95	0.95	1.00	1.00	0.95
0.06	0.35	0.90	0.95	0.95	0.95	0.90	0.95	1.00	0.95	0.95	0.95	0.95	0.95	0.95
0.07	0.60	0.80	0.90	0.95	0.95	0.95	1.00	0.95	0.95	0.95	1.00	0.95	0.95	1.00
0.08	0.90	0.95	0.95	0.90	1.00	0.95	0.90	0.95	0.95	0.95	0.95	0.95	1.00	1.00
0.09	0.90	0.90	0.90	0.95	0.95	0.95	1.00	1.00	1.00	0.95	0.95	0.95	1.00	0.95

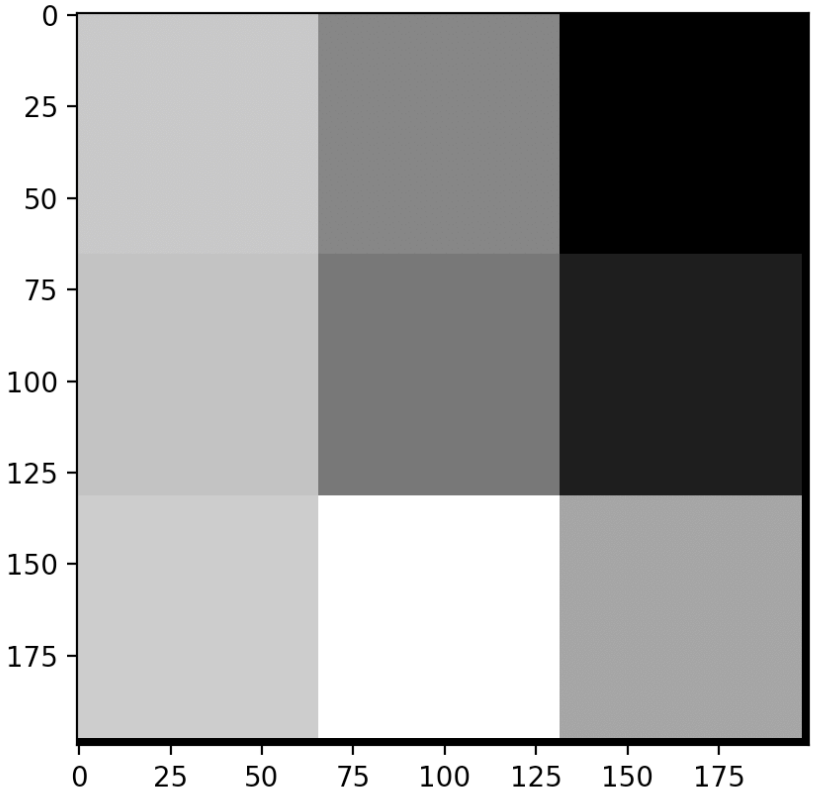
Low Learning Rate:

For this case, I am using the learning rate of 0.001, Relu function for my activation function and 500 epochs. The final accuracy is lower than 90%. The reason for this is that when the learning rate is low, it will take longer for the classifier to get to the optimal set of weights and biases. Therefore, in the case of my tests, since I'm only using 500 epochs, it cannot reach the optimal solution.

Figure 4: loss, accuracy and weights for a very low learning rate



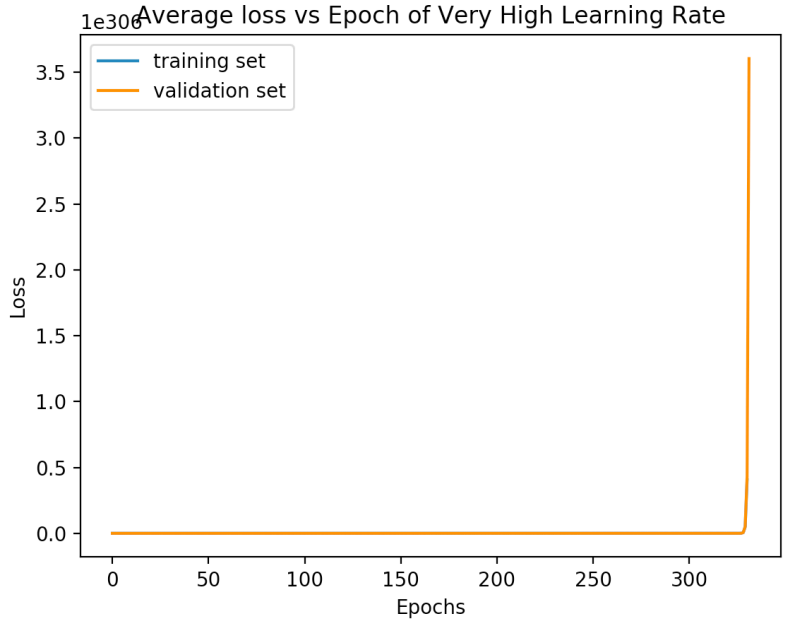


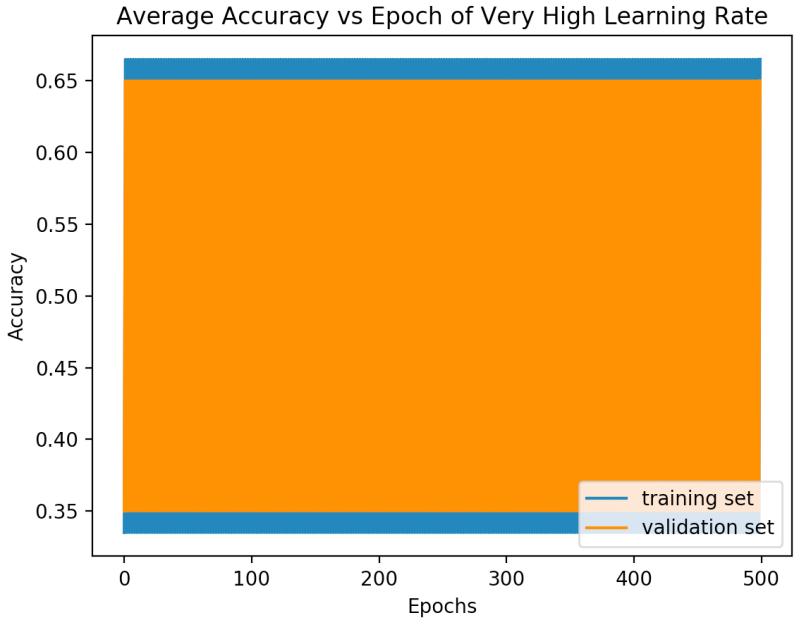

0
25
50
75
100
125
150
175
0
25
50
75
100
125
150
175

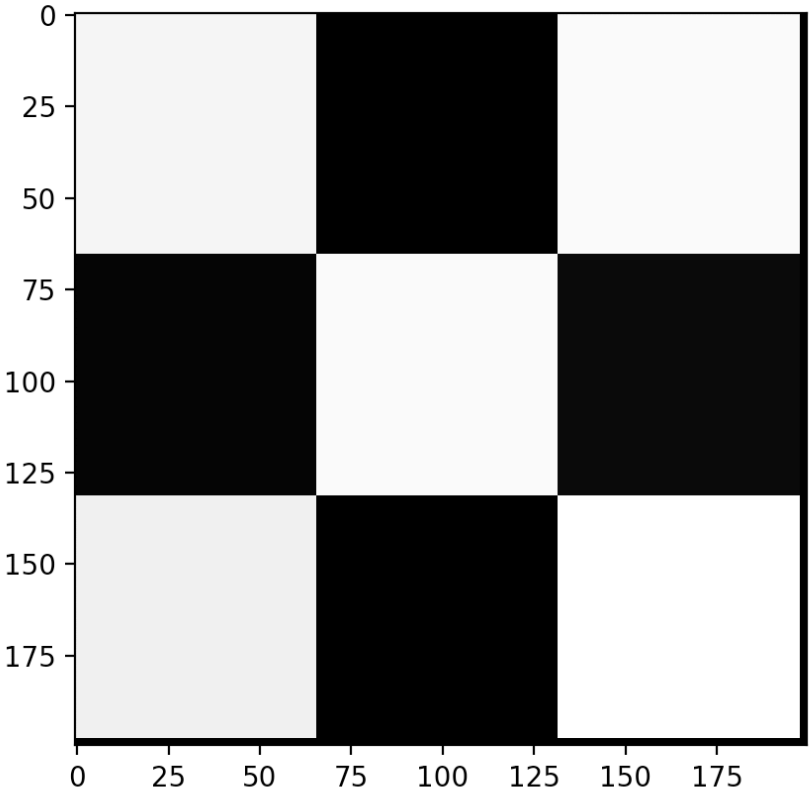
High Learning Rate:

For this case, I am using the learning rate of 0.5, linear function for my activation function and 500 epochs. In this case, since the activation function is very large, each time during gradient descent, the adjustments for weights and biases would overshoot, causing the accuracy to oscillate, which produced the pattern as shown in figure 5.

Figure 5: loss, accuracy and weights for a very high learning rate



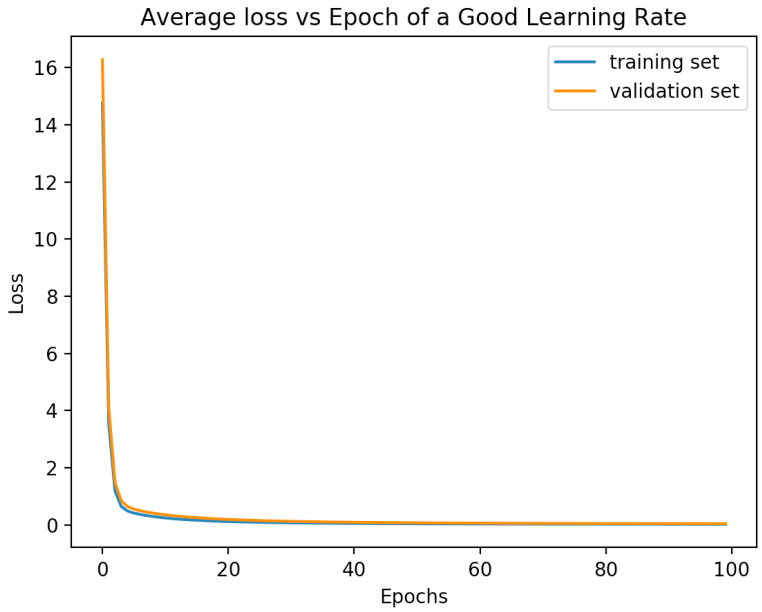


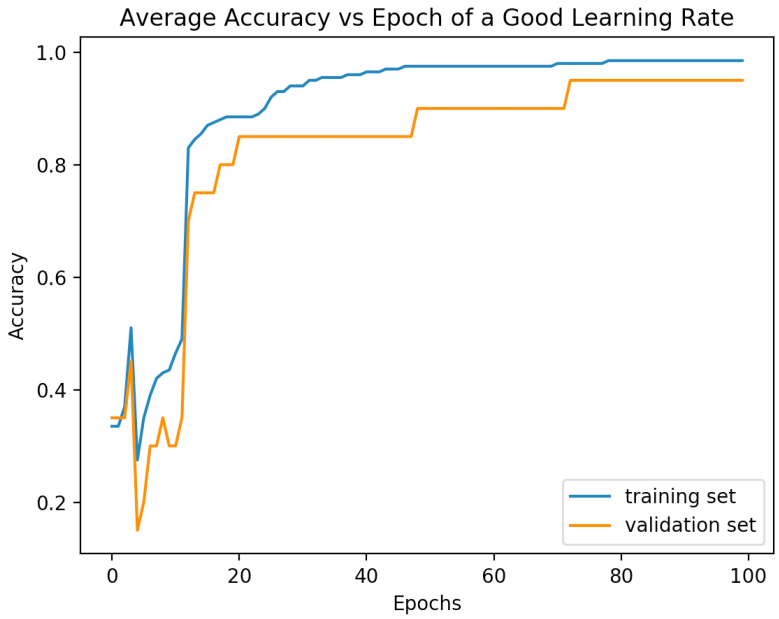

0
25
50
75
100
125
150
175
0
25
50
75
100
125
150
175

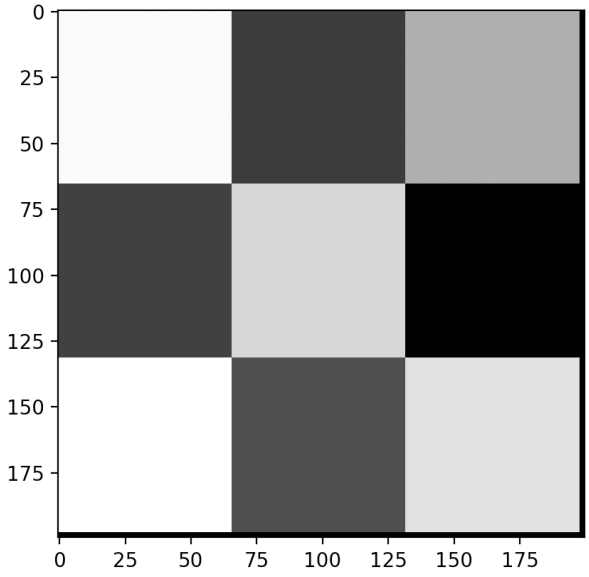
Good Learning Rate:

For this case, I am using the learning rate of 0.07, Relu function for my activation function and 100 epochs. The results are shown below.

Figure 6: loss, accuracy and weights for a good learning rate



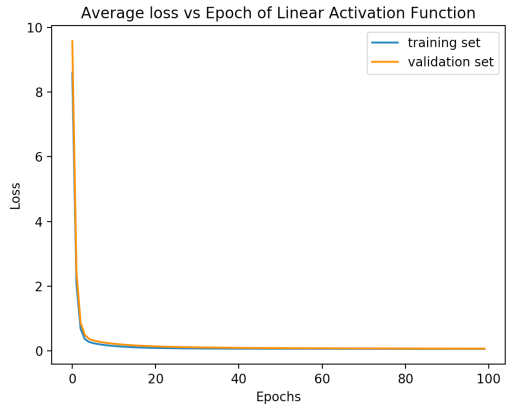


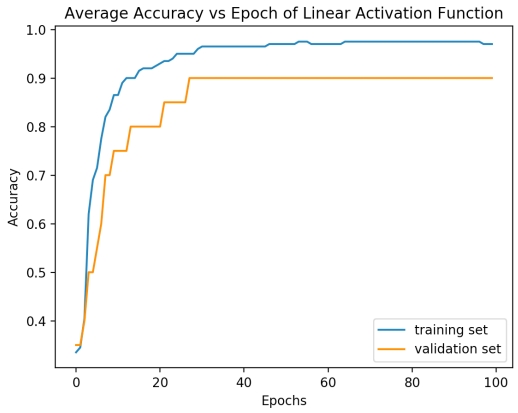

0
25
50
75
100
125
150
175
0
25
50
75
100
125
150
175

Three Activation Functions:

For this case, I am using the learning rate of 0.07 and 100 epochs. Figure 6 above shows the two graphs for the Relu activation function. Figure 7 and 8 shows the graphs for the linear and sigmoid activation function respectively. One thing to note is that since the sigmoid function is a modifies the weight a low slower, it is unable to alter the weight enough the improve the accuracy after only 100 epochs.

Figure 7:





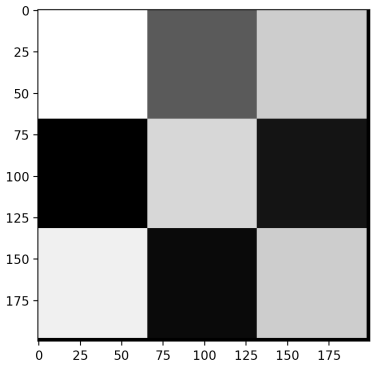


Figure 8: