Sudarshan Pillai

11/2/20

Ali Minai-EECE 6036

HW # 3

* 1. **System Description**

Number of hidden layers - 1

Number of hidden neurons – 100

Learning rate – 0.01

Momentum – 0.01

Output thresholds – 1 is >= 0.75, 0 is < 0.25

Hidden & Output layer activation - Sigmoid

Rule for choosing initial weights – random values between range of -1 to 1

Epochs - 200

Criterion to stop training – if loss function is less than 0.01

* 1. **Results**



Figure 1.1 – Hit rate error of the training set

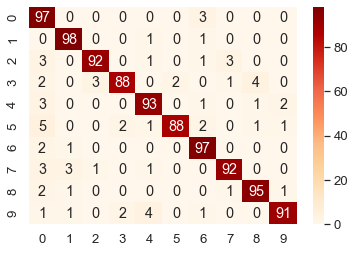


Figure 1.2 – Confusion matrix created from the test set

* 1. **Analysis of Results**

The learning rate was determined by trial and error after running through all the epochs and the final value which gave the best results for this algorithm was *lr = 0.01.* The momentum used for this algorithm after trial and error was *momentum = 0.*01, and the number of hidden layers neurons used was 100. Figure 1.1 shows the overall performance of the network over 200 epochs, and it can be noticed that the error rate decreases as the training continues and eventually stays constant around the 140th epoch. This was due to the condition that the training stops once it reaches an error less than 1%. From Figure 1.2, we can see that the model did a good job on classifying all the numbers. However, it performed the best on predicting the 1s which is probably due to the fact that the shape of a 1 is just a line.

**2.1 System Description**

Number of hidden layers - 1

Number of hidden neurons – 100

Learning rate – 0.01

Momentum – 0.01

Output thresholds – 1 is >= 0.75, 0 is < 0.25

Hidden & Output layer activation - Sigmoid

Rule for choosing initial weights – random values between range of -1 to 1

Epochs - 200

**2.2 Results**

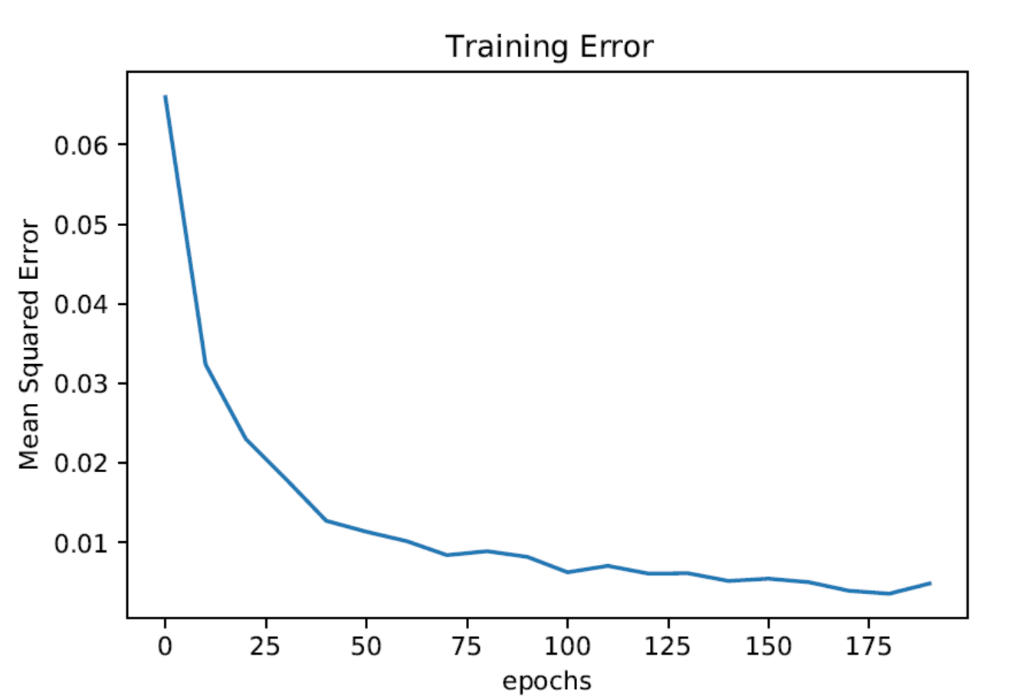


Figure 2.1 – Error on the training set

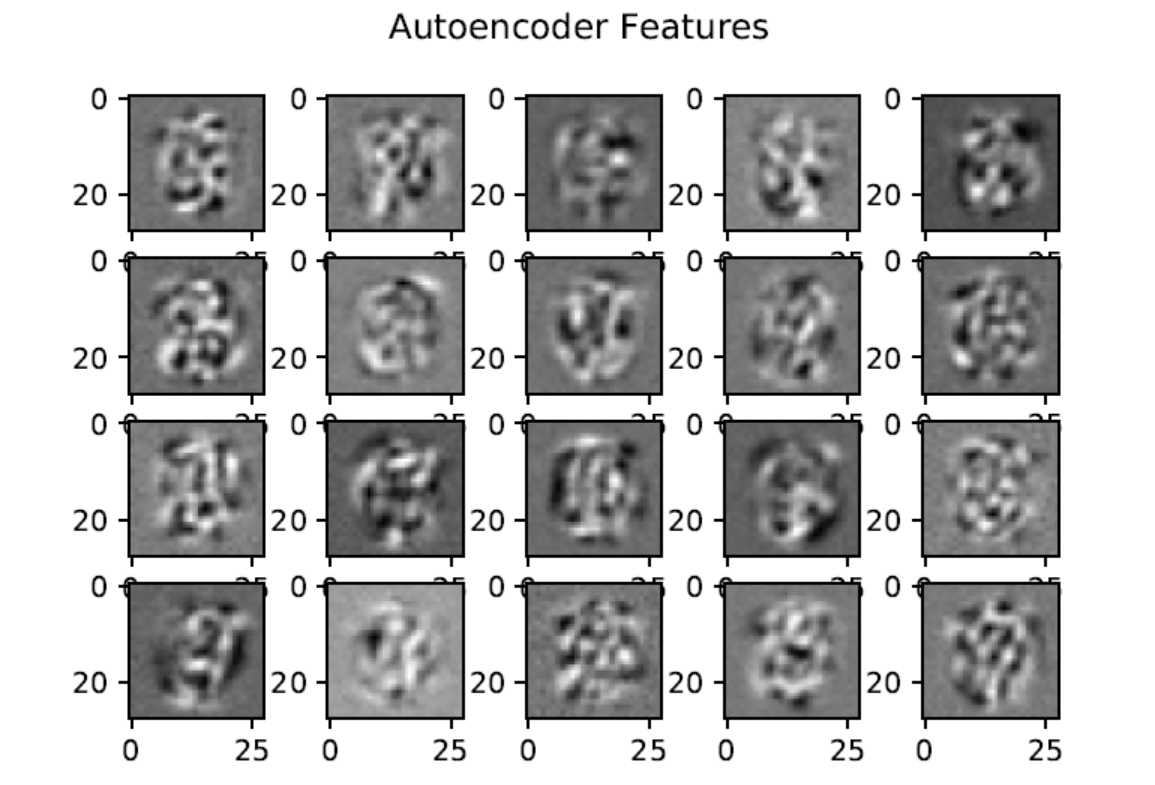
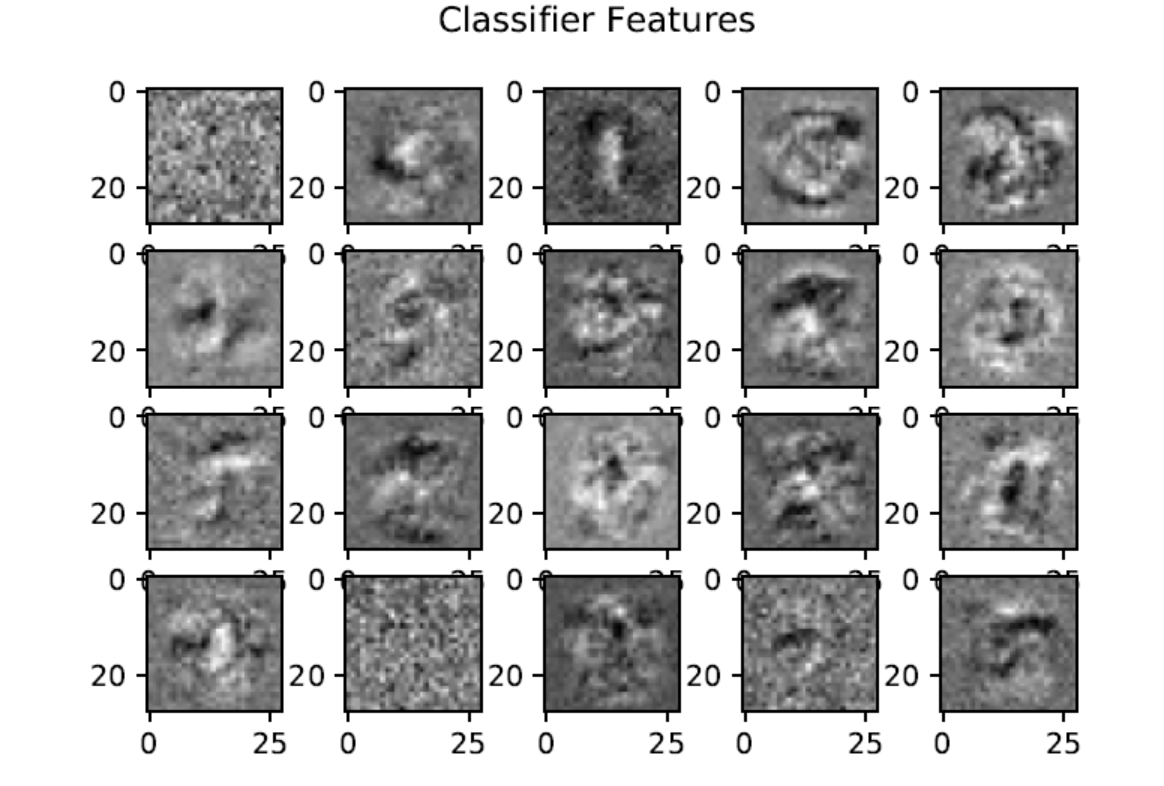
**2.3 Features**

Figure 2.2 – autoencoder features for 20 random neurons from the hidden layer

Figure 2.3 – classifier features for 20 random neurons from the hidden layer

With figure 2.3 and 2.4, we can see that the hidden layer features for both of these algorithms are producing some sort of random pattern which is not interpretable.

**2.4 Sample Outputs**

Figure 2.4 – Upper image is true values and bottom image is the image produced from autoencoder

**2.5 Analysis of Results**

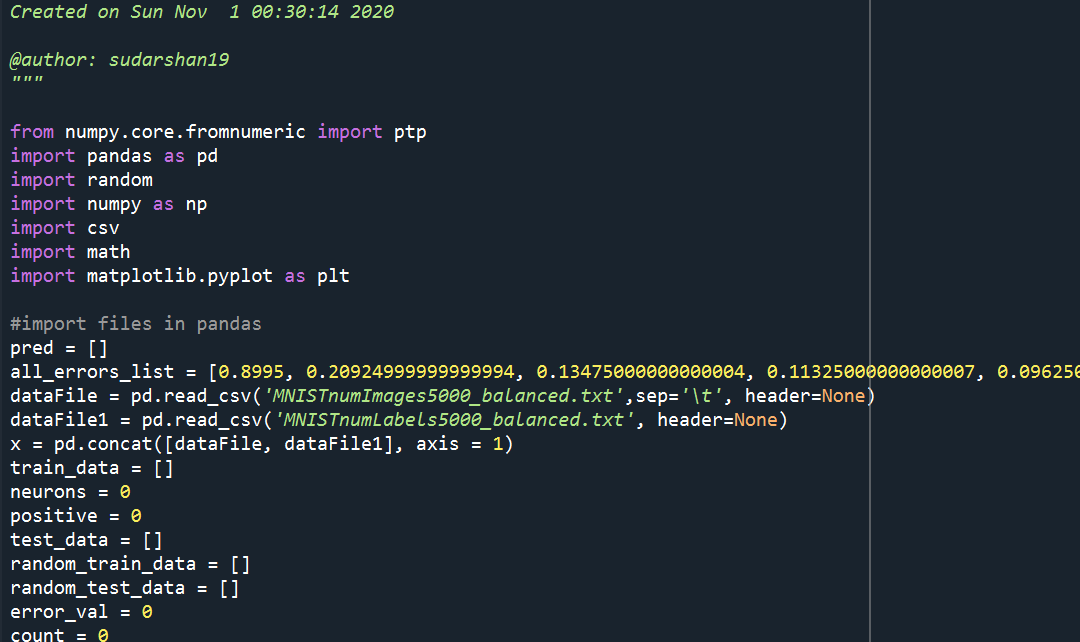
The learning rate was determined by trial and error after running through all the epochs and the final value which gave the best results for this algorithm was *lr = 0.01.* The momentum used for this algorithm after trial and error was *momentum = 0.*01, and the number of hidden layers neurons used was 100.Figure 2.1 shows the overall performance of the network over 200 epochs, and it can be noticed that the error rate decreases as the training continues and eventually stays constant around the 140th epoch.

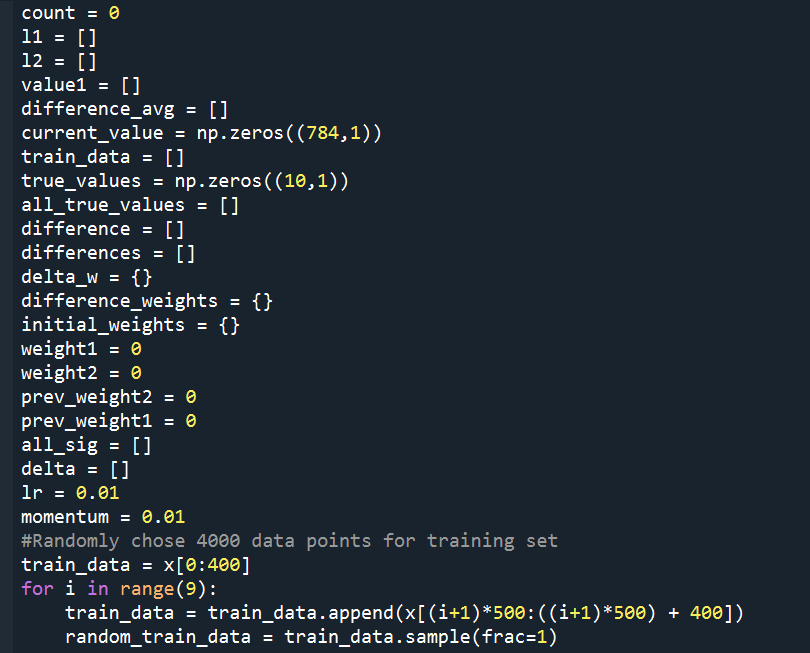
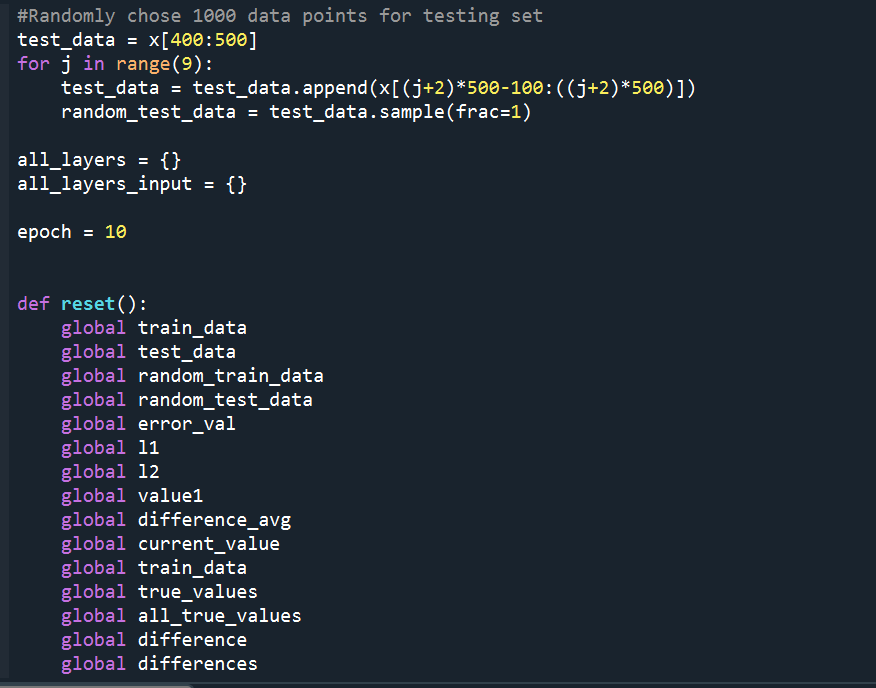
Figure 2.1 gives the error function for training set per epoch (200 epochs total). We can see that the error function is getting better and better as the epoch increases and decreases at a constant rate.

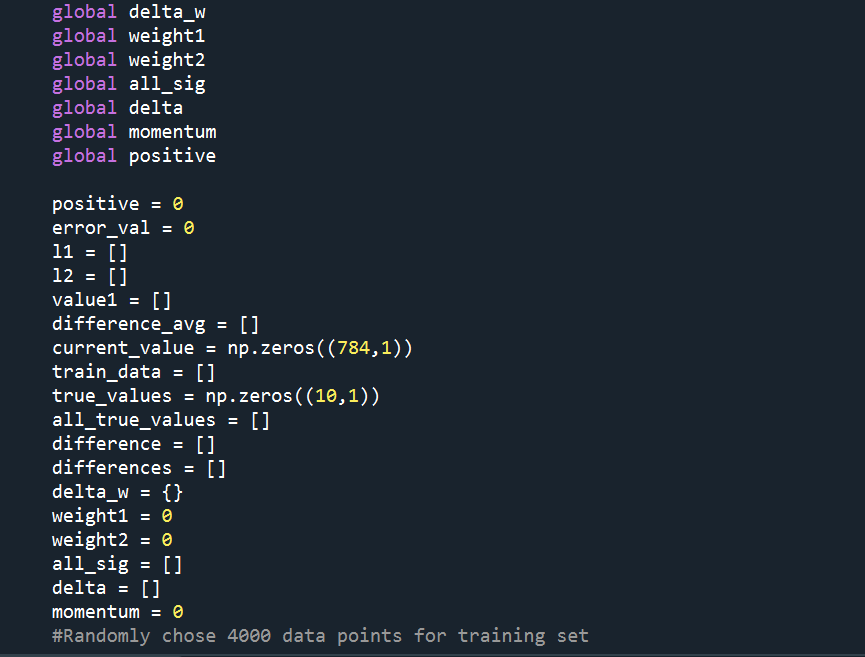
Figure 2.2 and 2.3 gives the feature images captured by the weight of random 20 neurons and for both the classifier and the autoencoder. The features of these algorithms are produced by the weights of the hidden layers. Finally, figure 2.4 shows the 8 random autoencoder outputs reconstructed to show the true image. The autoencoder algorithm is good enough to produce the original image with not much error.

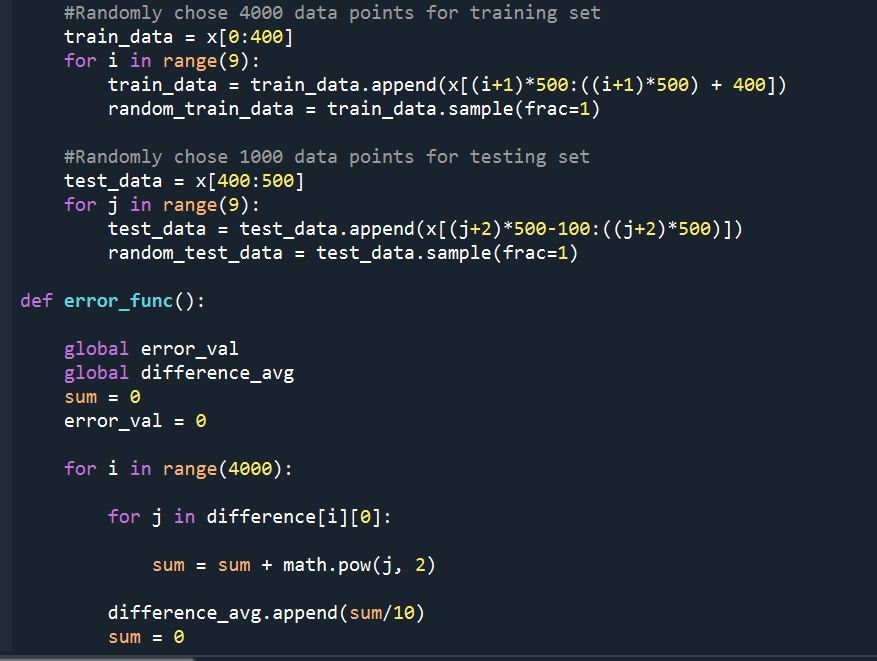
**3 Program Appendix**

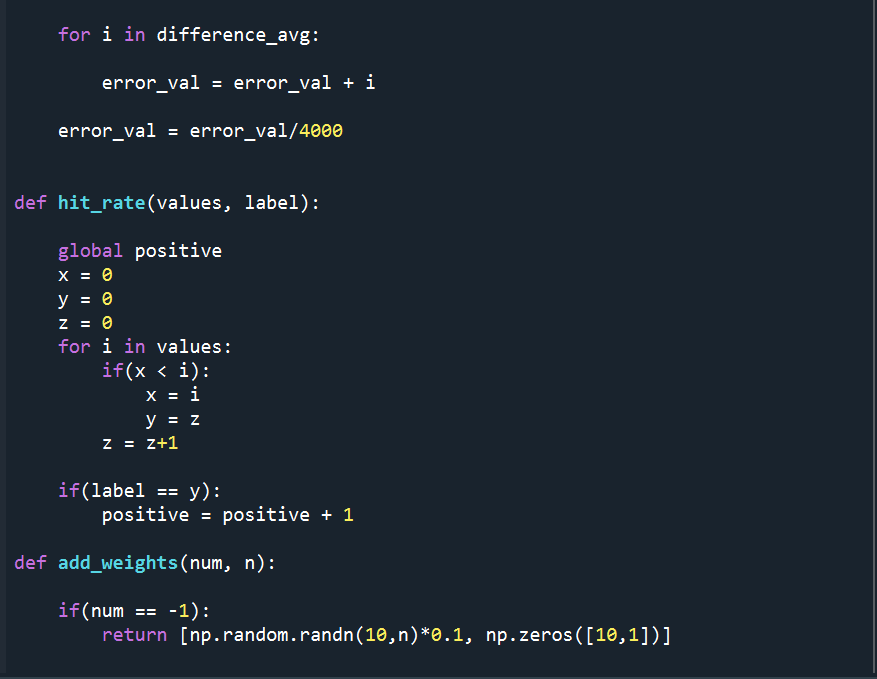
**3.1 Q1.py**

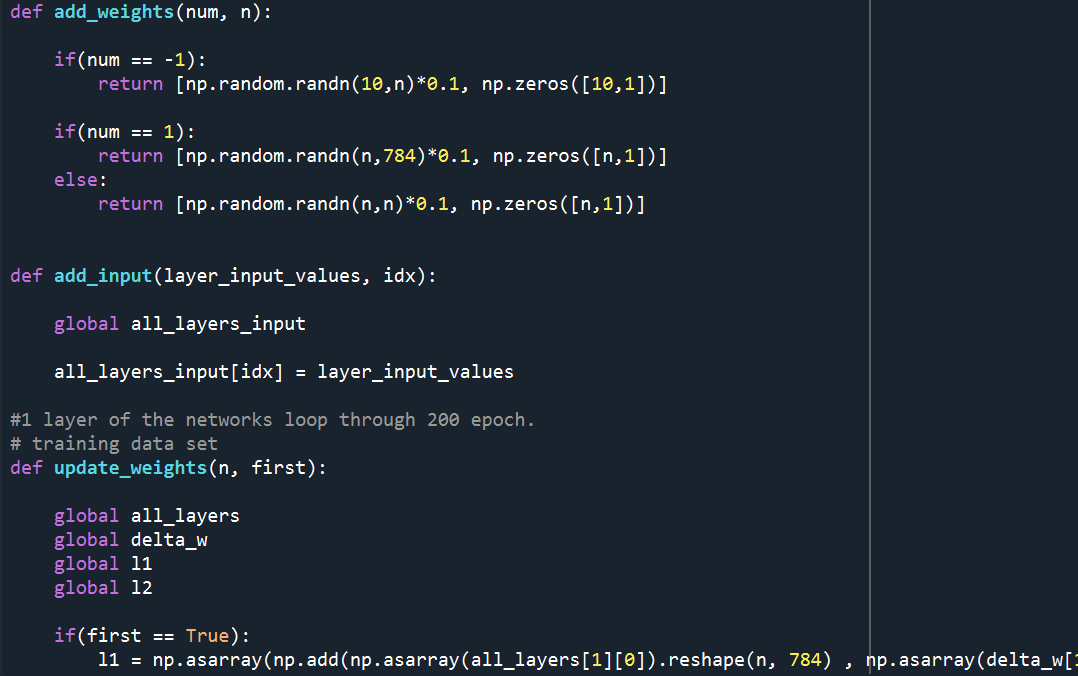




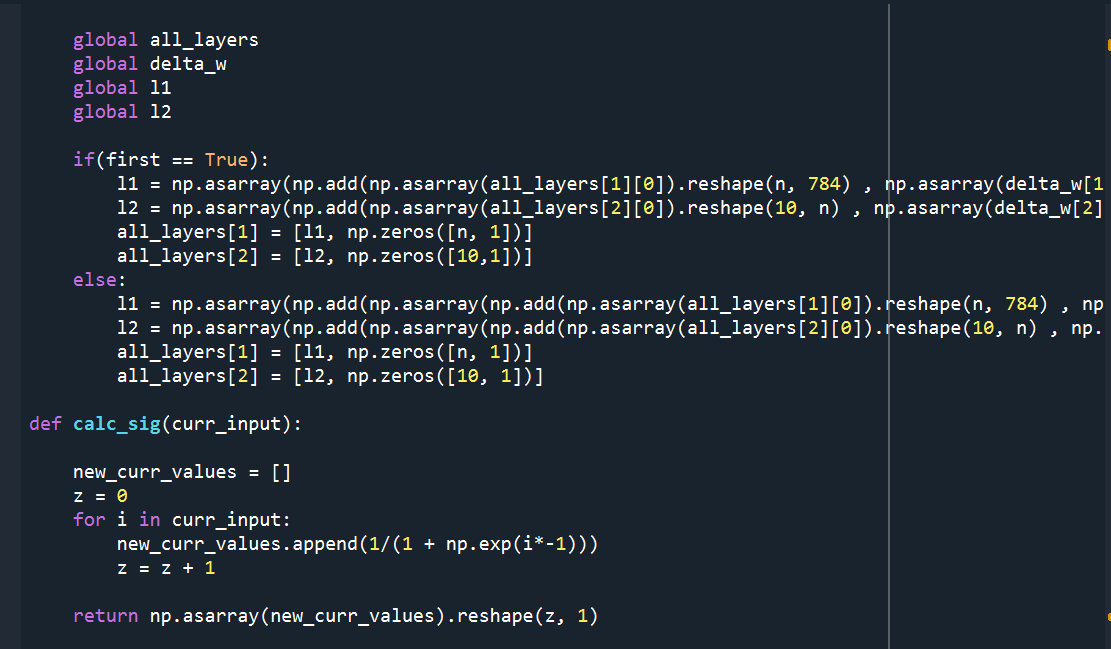
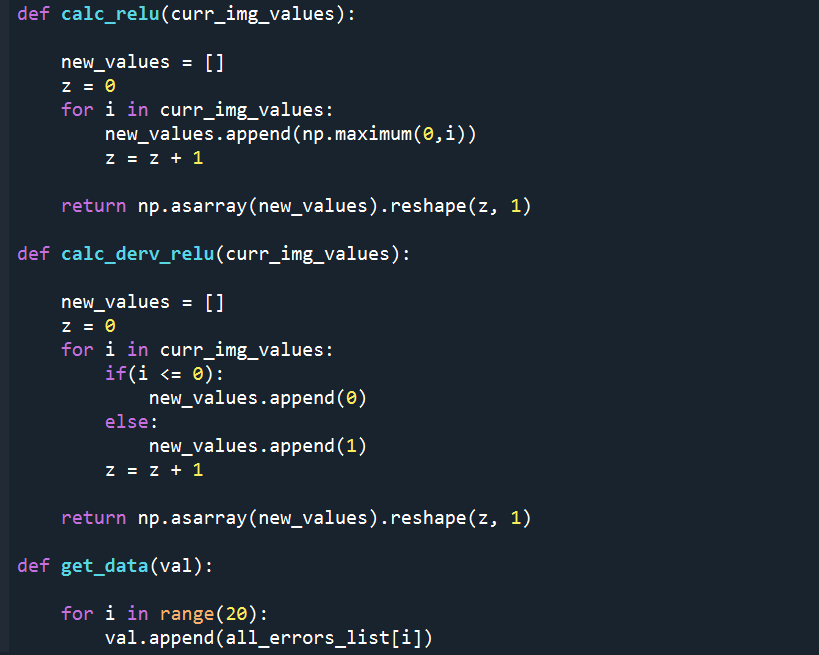
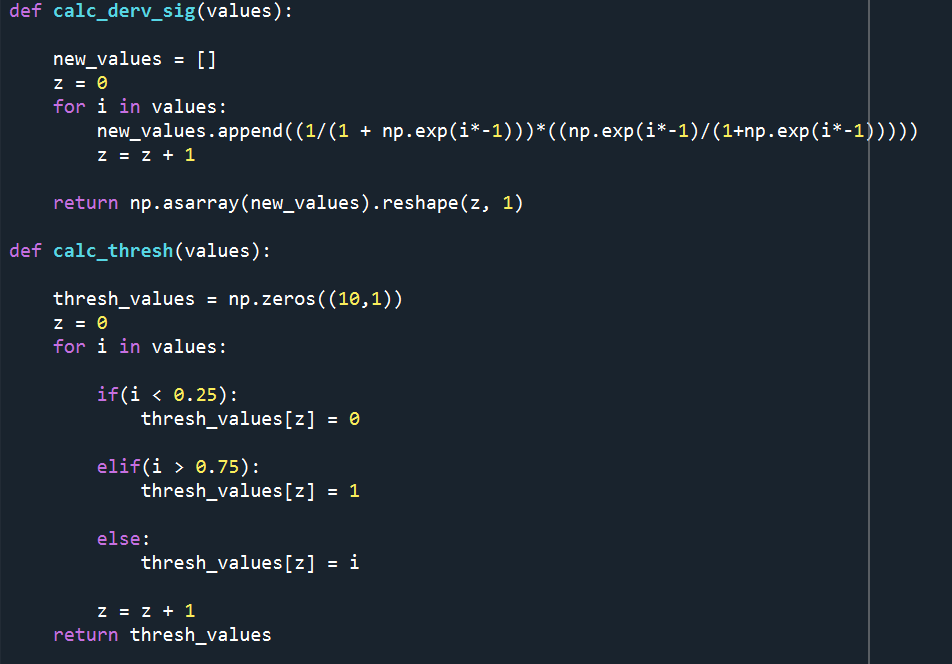


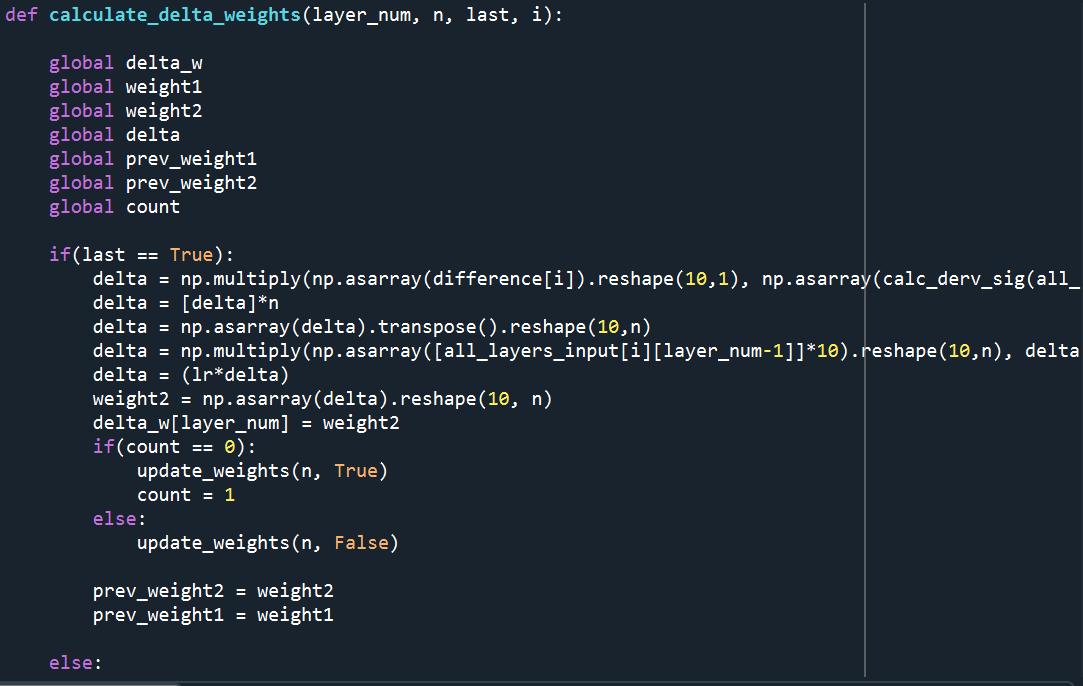
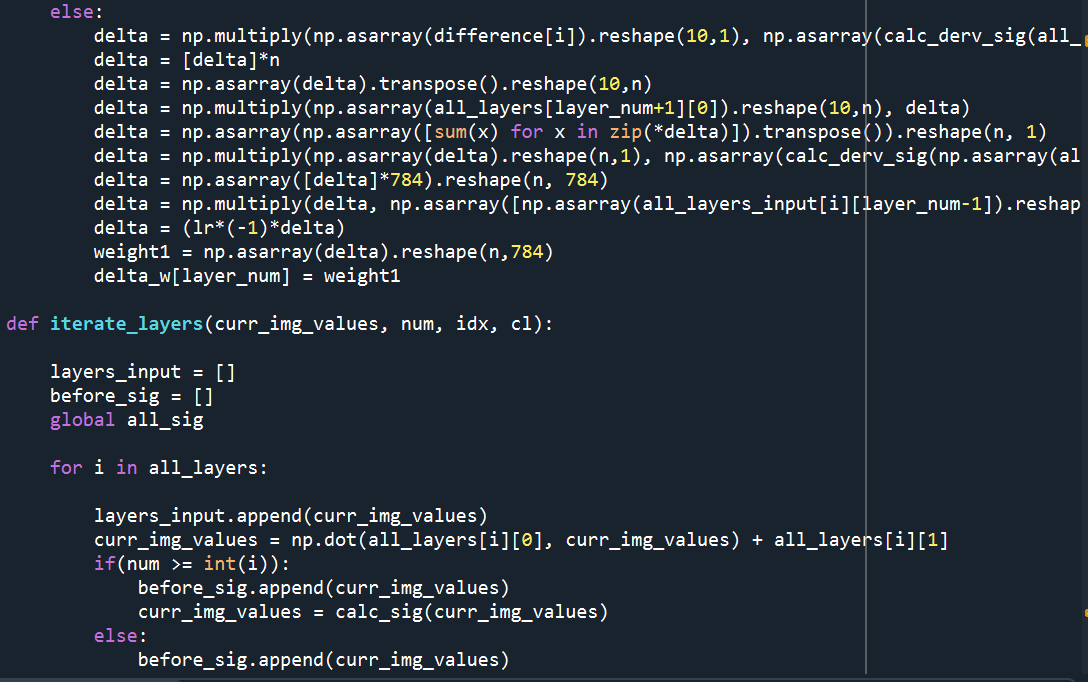
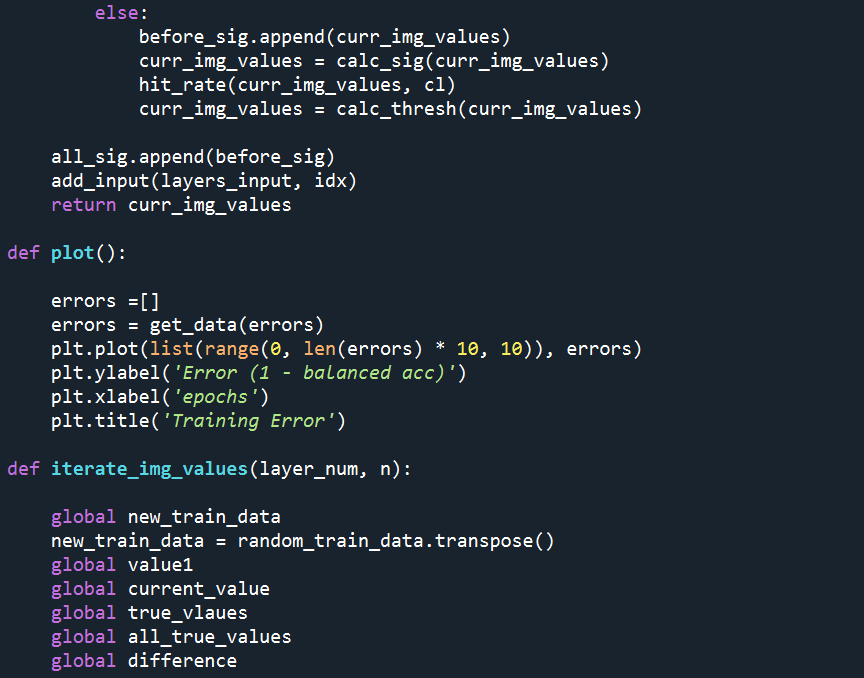


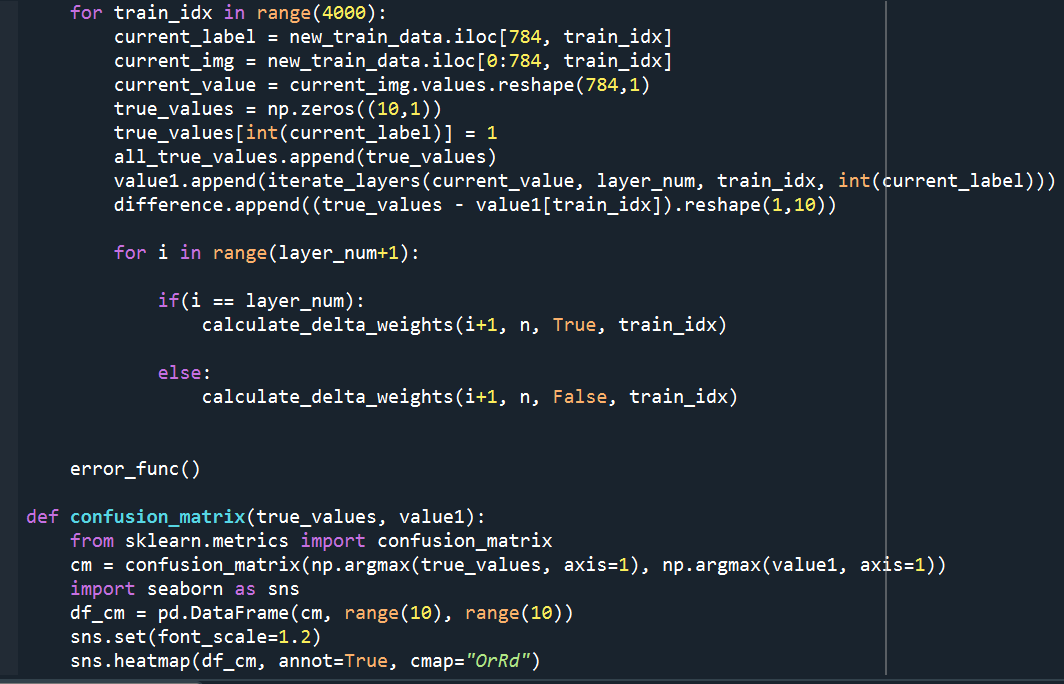
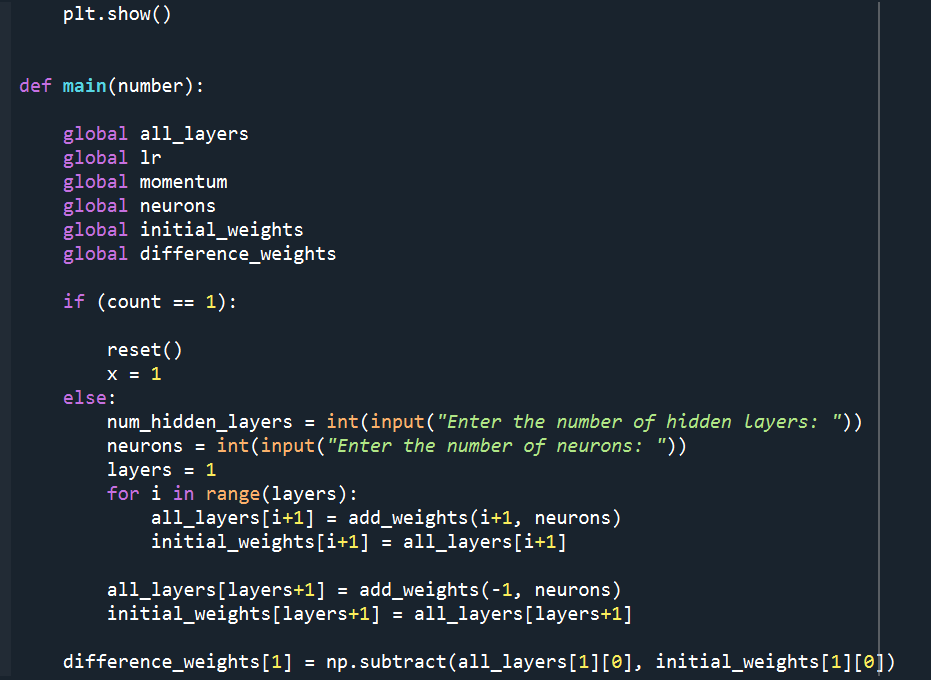


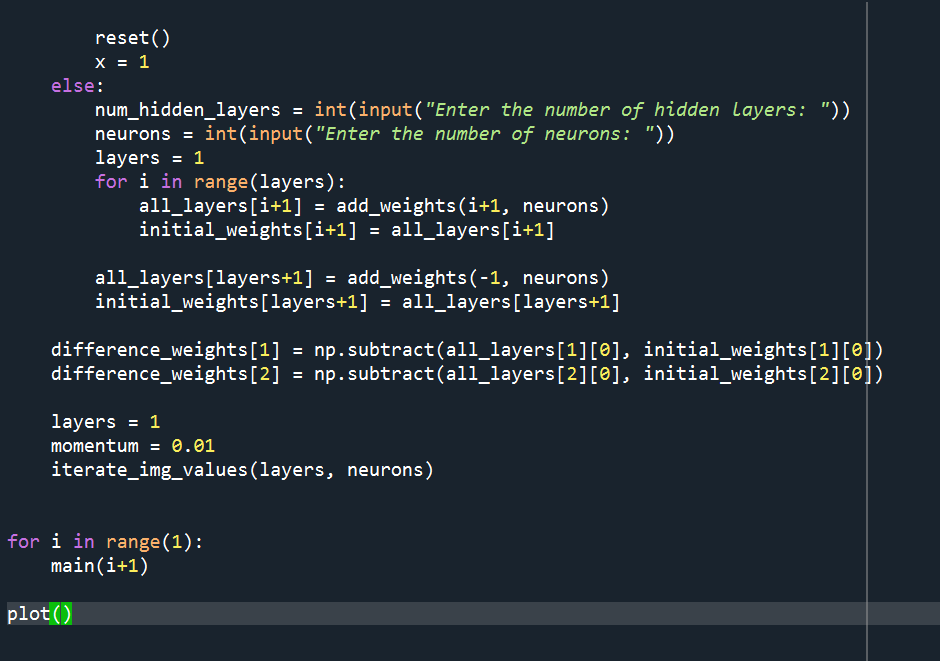
 /

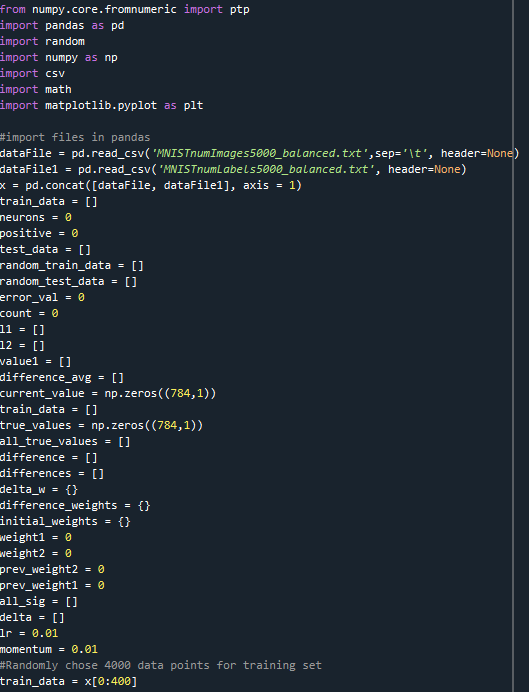
‘









**3.2 Q2.py**

