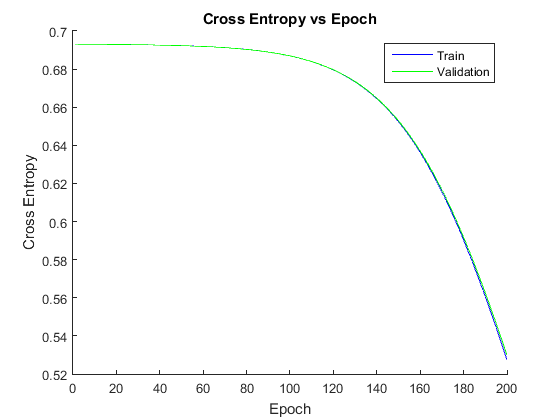
# 2 Neural Network

All classification error is expressed as decimals out of 1. To get the percent, multiply by 100%.

## 2.1 Basic generalization

The performance, in this case the cross-entropy, of the validation set plateaus after 700 epochs while the cross-entropy of the training set continues to decrease. This indicates that after around 700 epochs, the training is no longer generalizing and may be overfitting.

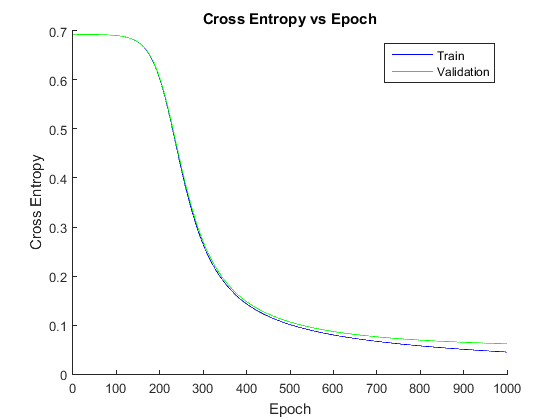
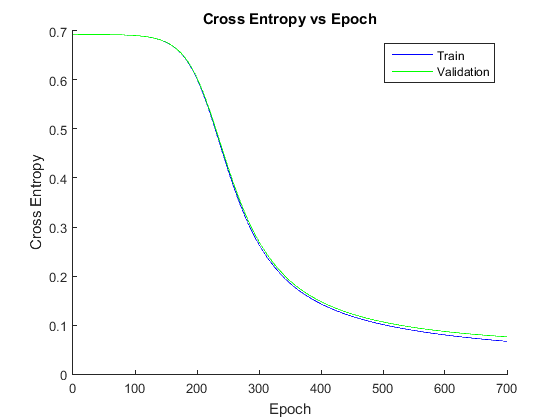
## \\SRVA\Homes$\chenleo5\Downloads\100.png



100 epochs compared to 200 epochs

It can be seen that in the first 200 epochs, the cross-entropy for validation and training are almost identical.

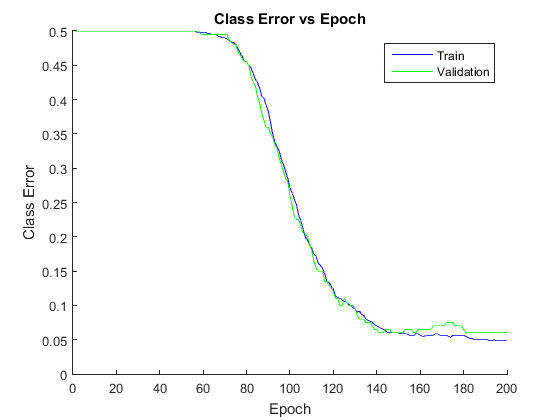
However, after around 700 epochs, the two curves visibly diverge, with the validation remaining relatively constant while the training decreases.



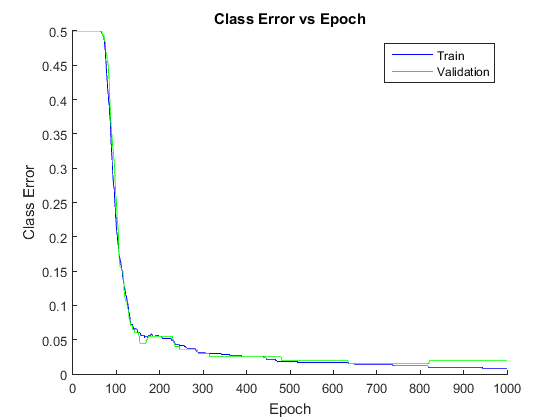
700 epochs compared to 1000 epochs

## 2.2 Classification error

In the first 50 epochs, there is no convergence at all. This could be due to the poor initial values chosen, so it takes some time to start converging.



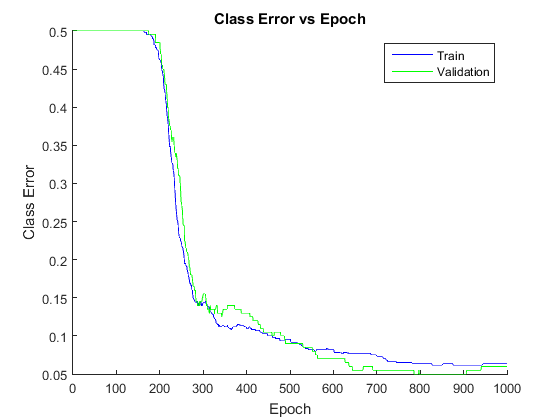
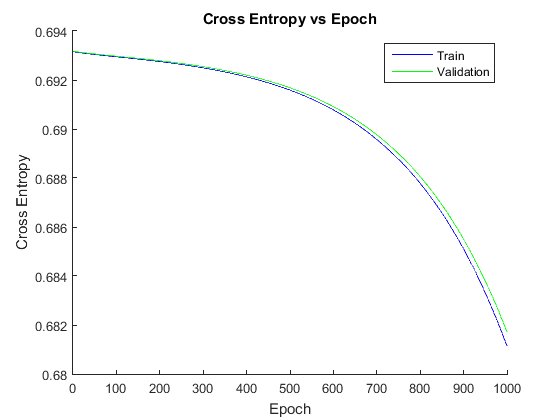
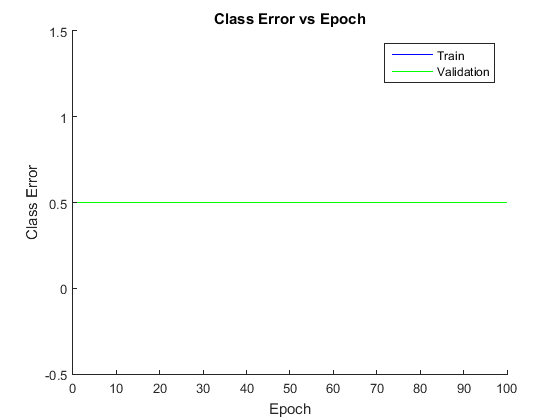
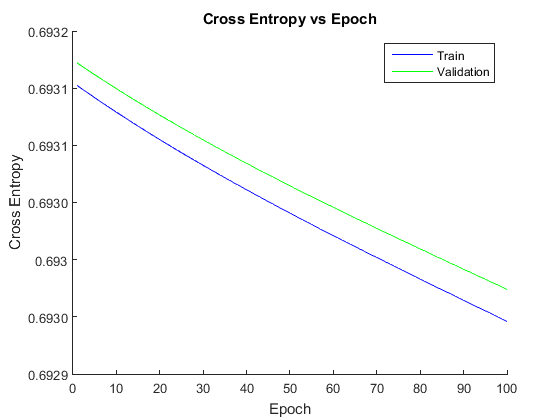
It can also be seen at after 700 epochs, the error for validation starts to increase while the training continues to decrease.



The fluctuations in the classification error is likely due the decision boundary at 0.5 (50%). There can be large changes in the probability, but if the probability does not cross the 0.5 threshold, no change can be seen. Likewise, a small change that pushes the probability over the threshold will cause a comparatively large change in the classification.

## 2.3 Learning rate

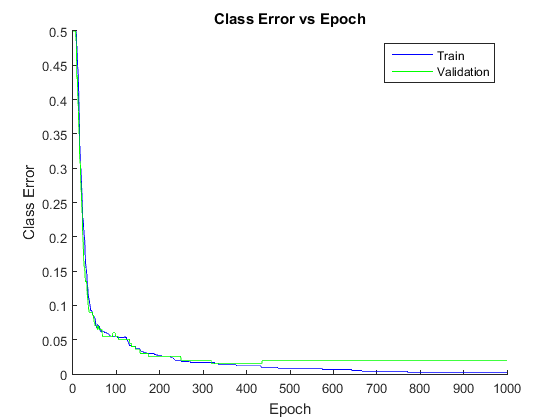
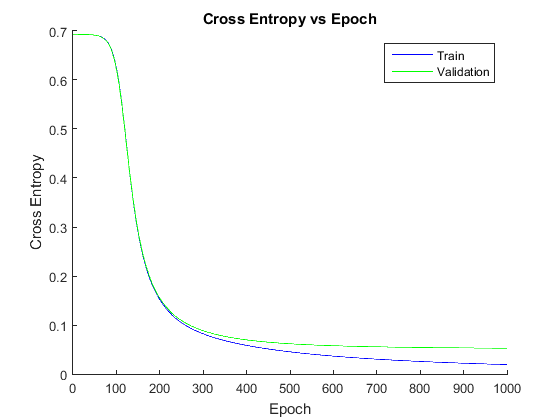
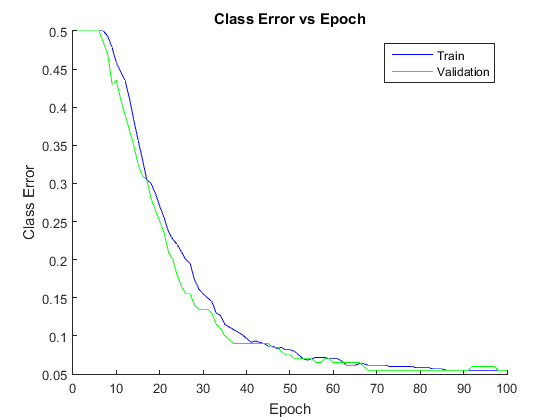
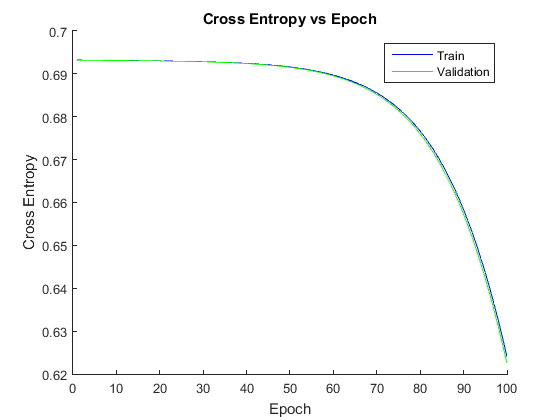
### Number of Hidden Units



0.01 learning rate: 100 and 1000 epochs

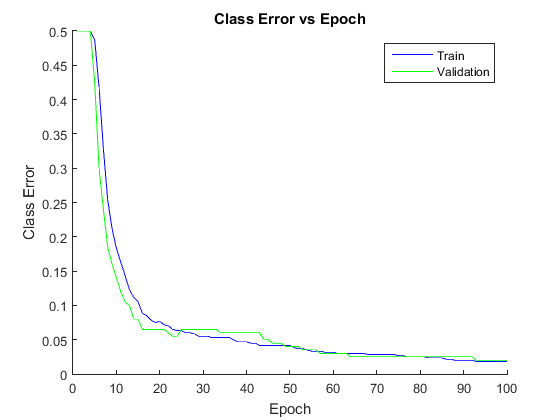
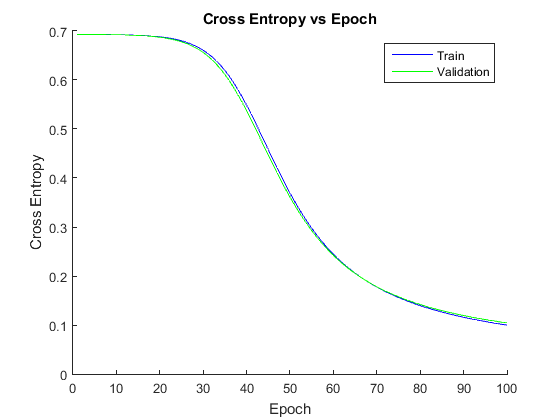
The convergence rate for 0.01 is visibly lower, taking almost 200 epochs before even starting to converge at all. This can be seen in the classification error which is not much better than random guessing.

The number of epochs before the validation starts to diverge, 900, is also higher than before. The cross-entropy does not even seems to converge after 1000 epochs, with training and validation almost the same.

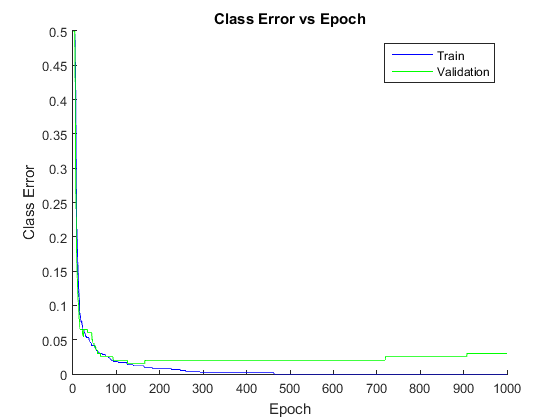
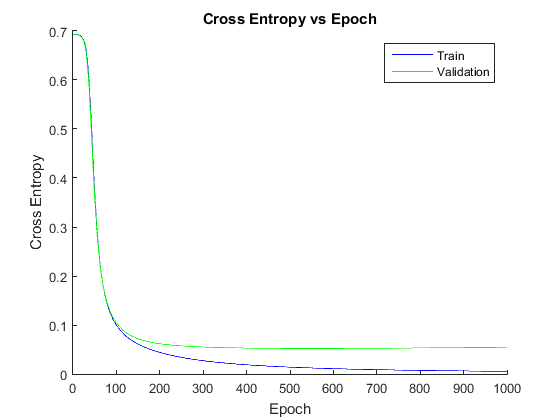


0.2 learning rate: 100 and 1000 epochs

The convergence rate for 0.2 is visibly faster, taking only 10 epochs before converging. The epochs taken to converge is also lower, being 400 epochs.

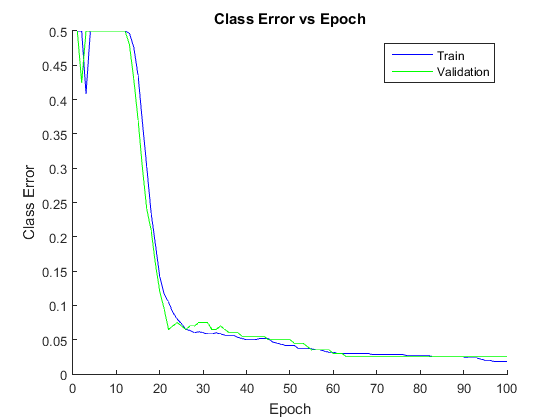
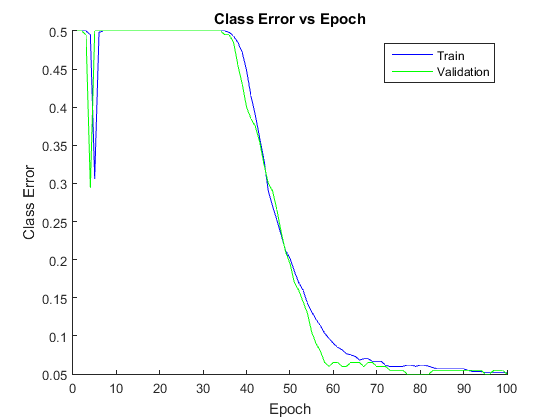


0.5 learning rate: 100 epochs



0.5 learning rate: 1000 epochs

The convergence rate for 0.5 is even faster, taking only 5 epochs before converging. The epochs taken to converge is very low; after only 150 epochs the classification and cross-entropy for the validation set starts to diverge.

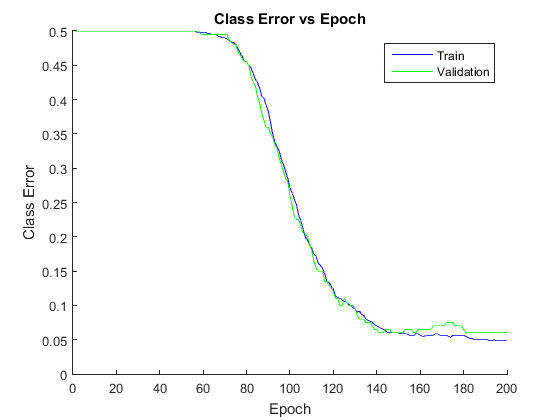
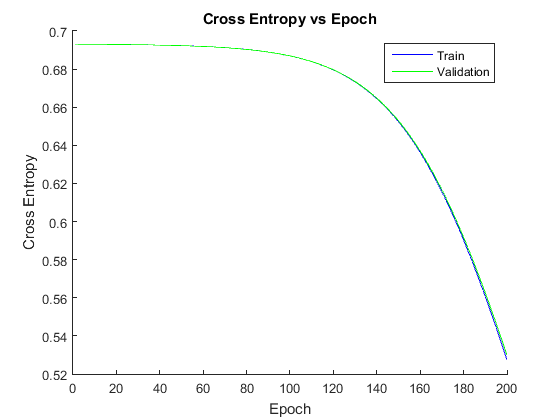


0.2 and 0.5 learning rate being misled

While a higher learning rate converges faster, there is a higher chance of being misled and exhibiting oscillatory behavior. The higher the learning rate, the higher the likelihood of oscillating.

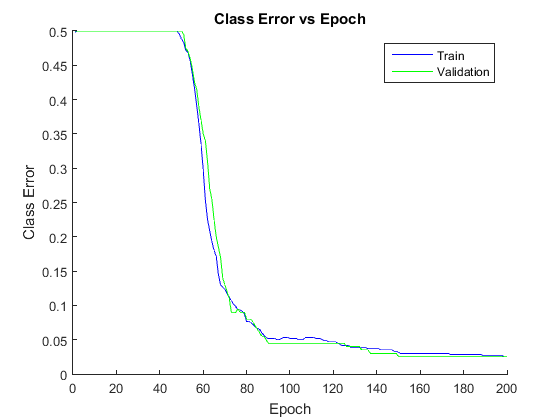
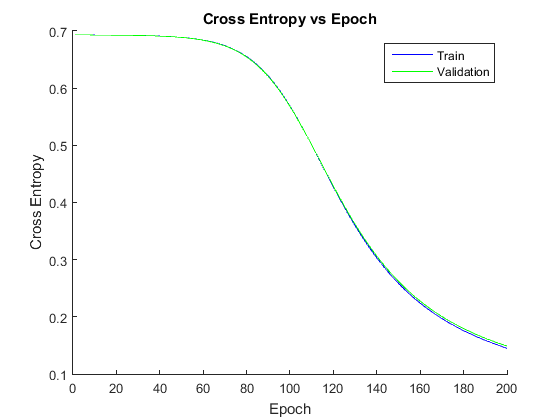
### Momentum

For this section, the learning rate has been returned to 0.1.



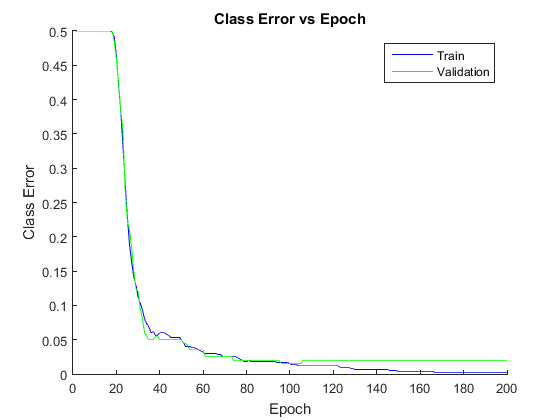
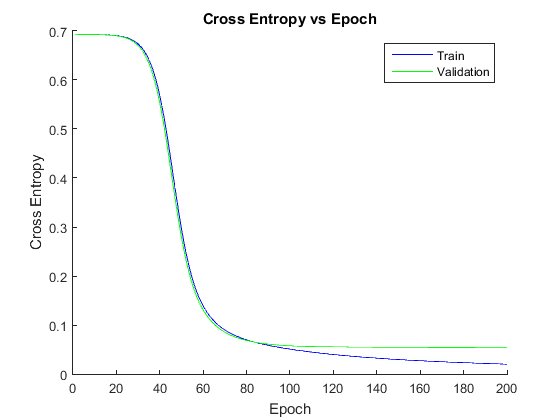
0.0 momentum: 200 epochs

With no momentum, not only does it take a while to converge, there is a higher chance of being misled and oscillating. This can be seen at around 150-180 epochs, in the classification error for validation.



0.5 momentum: 200 epochs

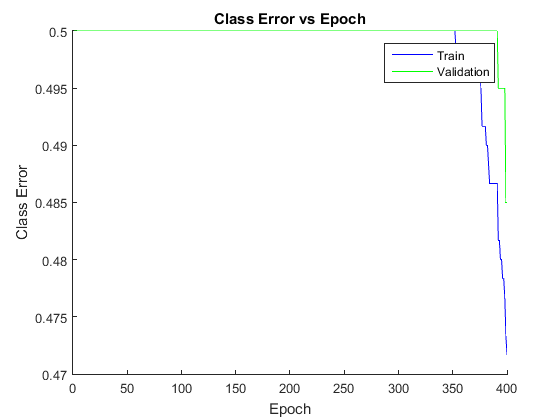
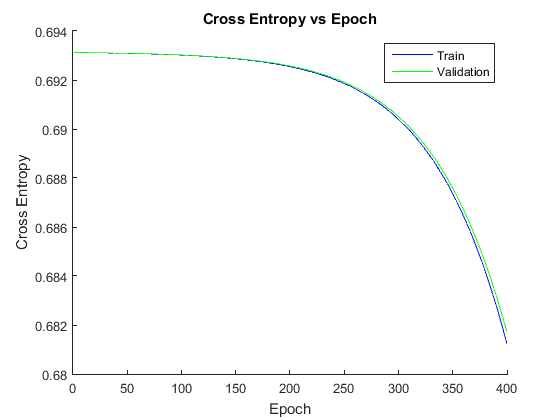
With 0.5 momentum, it can be seen from the cross-entropy and classification error that convergence happens faster. There is also less oscillations in both training and validation.



0.9 momentum: 200 epochs

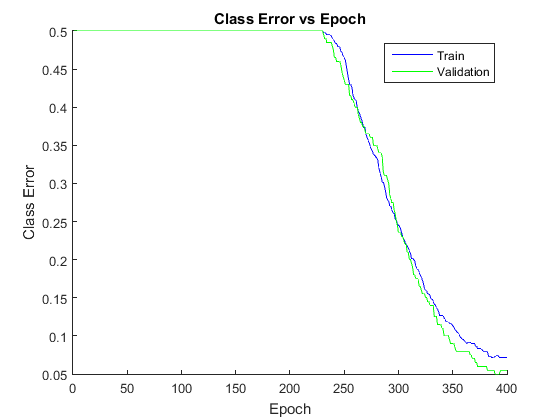
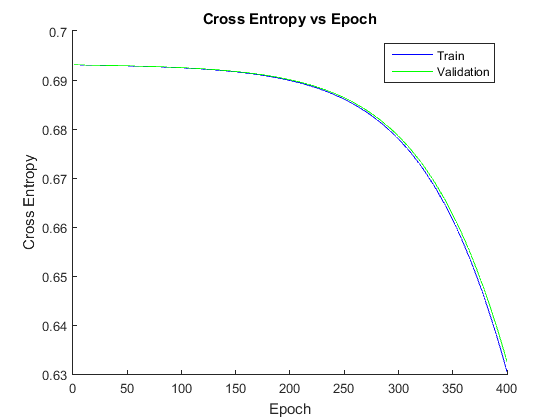
With 0.9 momentum, not only do the convergence happens even faster, the amount of time it takes before the it starts converging is also shorter; taking only 20 epochs as opposed to 50 from before. Of course, there is also less oscillations in both training and validation.

## 2.4 Number of hidden units



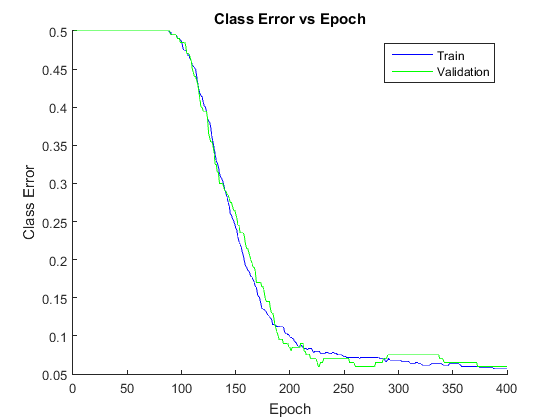
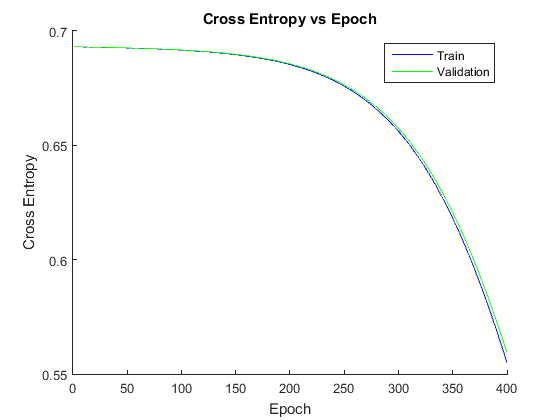
2 hidden units: 400 epochs

With less hidden units, the convergence happens slower. This can be seen in the classification error, where it takes 350 for either training or validation to start to converge.



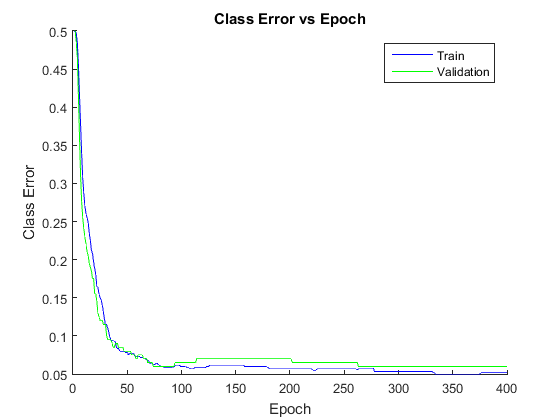
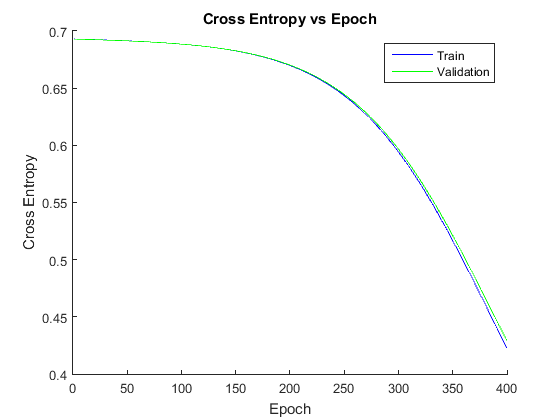
5 hidden units: 400 epochs

With 5 units, it can be seen from the classification error that convergence happens faster than 2 units. Starting at 250 epochs.



30 hidden units: 400 epochs

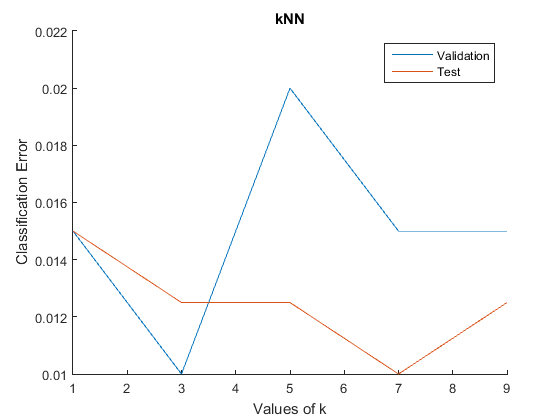
With 30 hidden units, the convergence speed is almost doubled. There is however, some oscillatory behavior in the validation.



100 hidden units: 400 epochs

With 100 hidden units, the convergence begins immediately and plateaus at 100 epochs. There is however, significant oscillations as it validation diverges at 100 then converges again at 300.

## 2.5 Compare k-NN and Neural Networks



Using a learning rate of 0.02, momentum to 0.5, and 30 hidden units for the neural network.

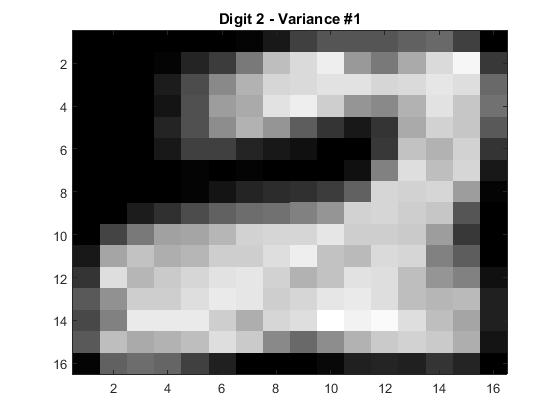
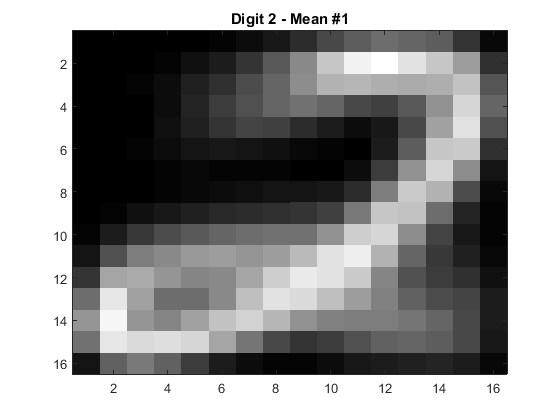
Cross-entropy for the test set is 0.109137, but there is no cross entropy for k-NN.

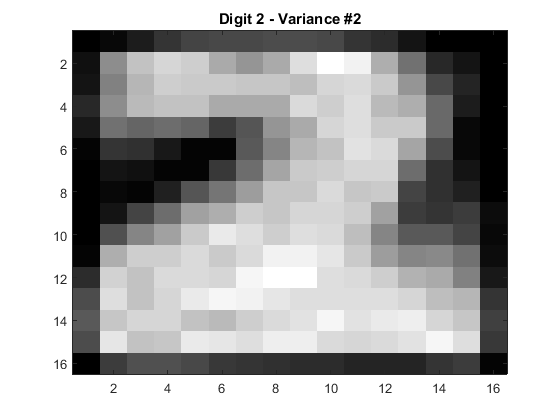
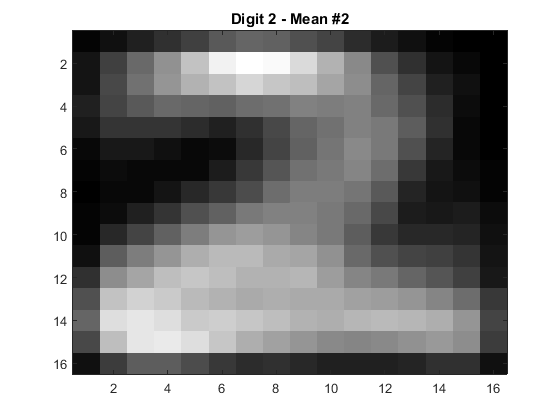
Classification error for the test set is 0.025. On the other hand, average of classification error for the test set is 1.25%. That’s half of the NN, which is at 2.5%.

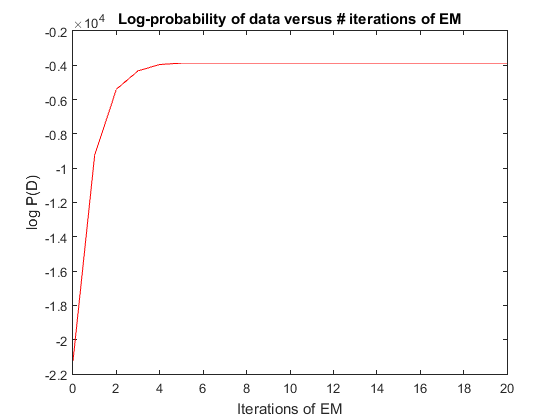
k-NN is better, but multiple k values needed to be tested for the best result. Meanwhile, NN generally converges given enough iterations.

# 3 Mixture of Gaussian

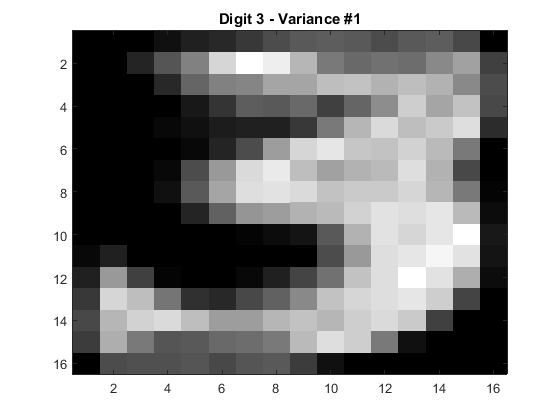
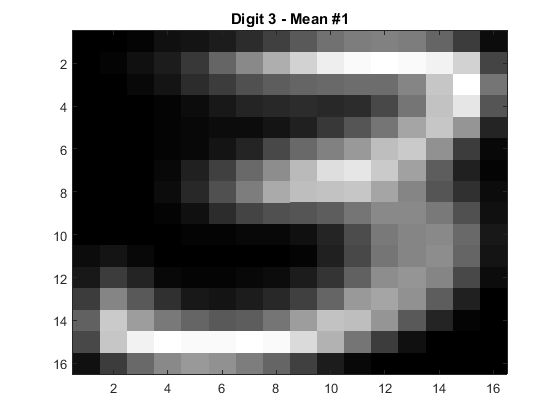
## 3.2 Training

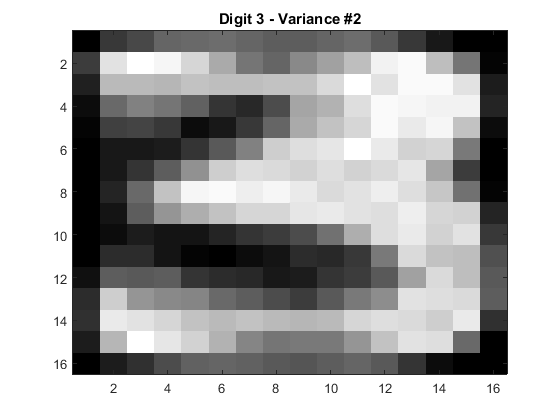
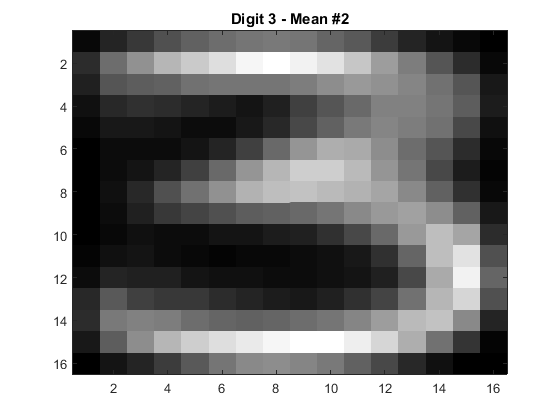


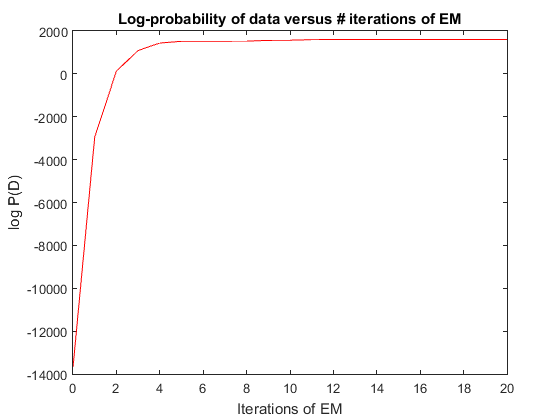




The **randConst** was set to 4 and the number of iterations to be 21 after much testing. Most of the convergence is finished after 8 iterations, so convergence is guaranteed with this model. Each mean is one distinct generalization of a handwritten two.

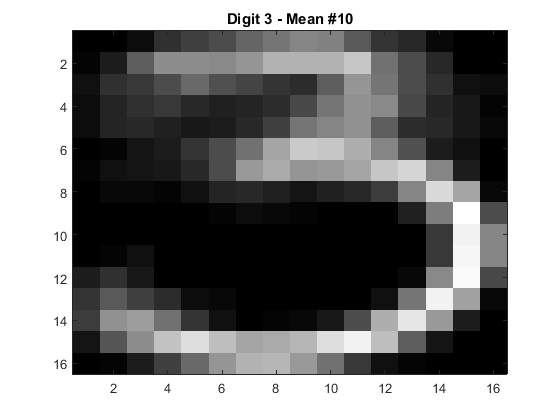
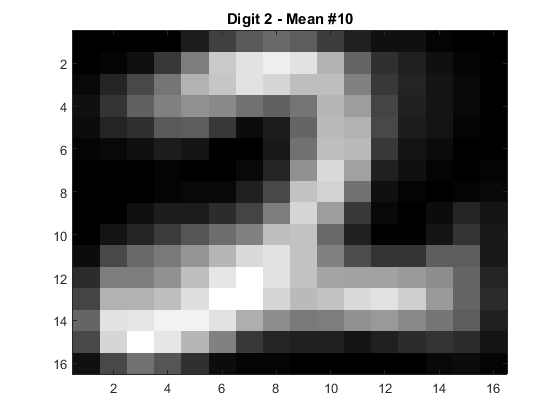


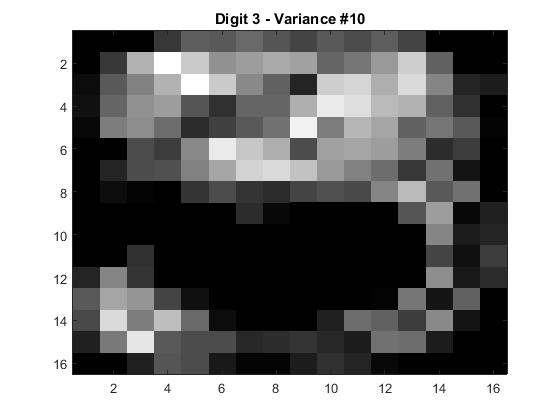
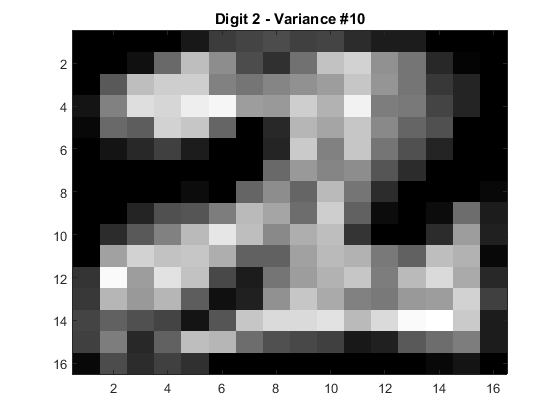


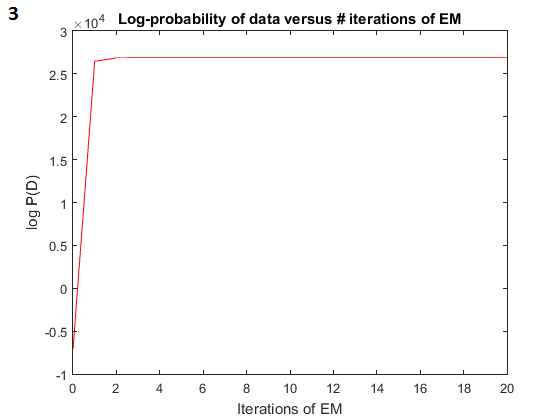
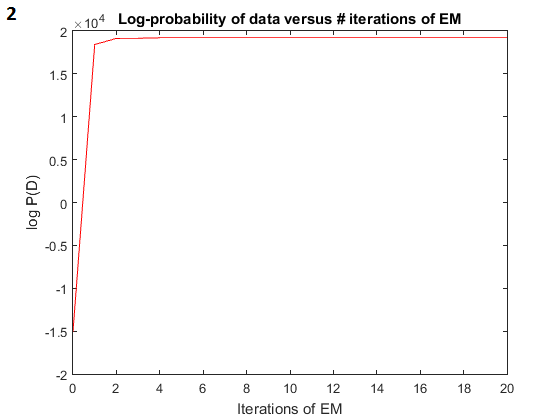


The **randConst** was set to 4 and the number of iterations to be 21 after much testing. Most of the convergence is finished after 10 iterations, so convergence is guaranteed with this model. Each mean is one distinct generalization of a handwritten three.

## 3.3 Initializing a mixture of Gaussians with k-means

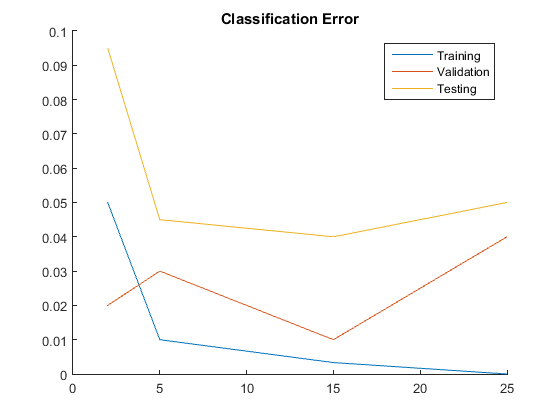
****

****

****

With k-means to initialize the clusters, the convergence happens faster, taking only 5 iterations for two and 6 iterations for three. The value converged to, is also higher compared to before.

## 3.4 Classification using MoGs



1. The error decreases as the number of clusters as there are more parameters to describe the data. More clusters being mixed means more complicated densities can be approximated. Trivially, with number of clusters equal to the number of the data points, zero is guaranteed.
2. The test set’s performance improves as the number of the cluster increase until 15, but diverges at 25 clusters. Hence, after a certain number of clusters, overfitting occurs.
3. To pick the best, I would test different number of clusters, run the test multiple times, and take the one with the best average results.

For the lowest error, I would pick 15 clusters for the model. This is because in the numerous run done, 15 rarely overfits and has better performance compare to 5 and 25.

## 3.5 Mixture of Gaussians vs Neural Network

For the MoG, 15 clusters was used with 21 iterations.

Results of MoG:

* Train = 0
* Validation = 0.04
* Test = 0.045

For the NN, learning rate is 0.02, momentum is 0.5, hidden units is 30, and 1500 epochs (15 *training\_nn* runs). The *training\_nn*, *init\_nn*, and *test\_nn* files were moved into the same folder as the MoG files.

Results of NN:

* CE = 0.079624
* CL = 0.0225

It can be seen that the NN performs better than the MoG, with an error of 2.25% as opposed to the 4.5% from the MoG.

Visualizing the weights as images, see the following pages, we get images which looks like a mixture of a two and a three. Comparatively, the means from the mixture model has a far more intuitive appearance (See Questions **3.2** and **3.3**). So, while the MoG performed worst, it would be easier for a human to check and understand the model.