Pattern and Speech Recognition WS2015-16
Exercise 9

Atanas Poibrenski(2554135), Marimuthu Kalimuthu(2557695), Furkat Kochkarov(2557017)

January 15, 2016

Neural Networks

Implementation of the NN

- See 'predict.m'. It returns a vector of dimension 400x1
- See 'trainNetwork.m'
- We use *rand* function in matlab for initializing the parameters using uniform distribution.
- Done. We use *randperm* in matlab for shuffling the data. Plots of the errors for different learning rates are as follows:

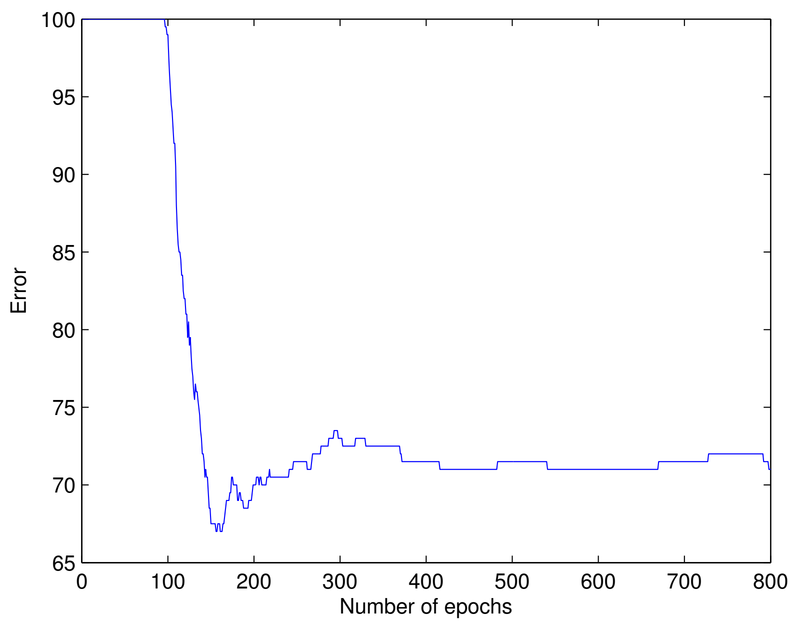

Figure 1: Learning rate 0.001

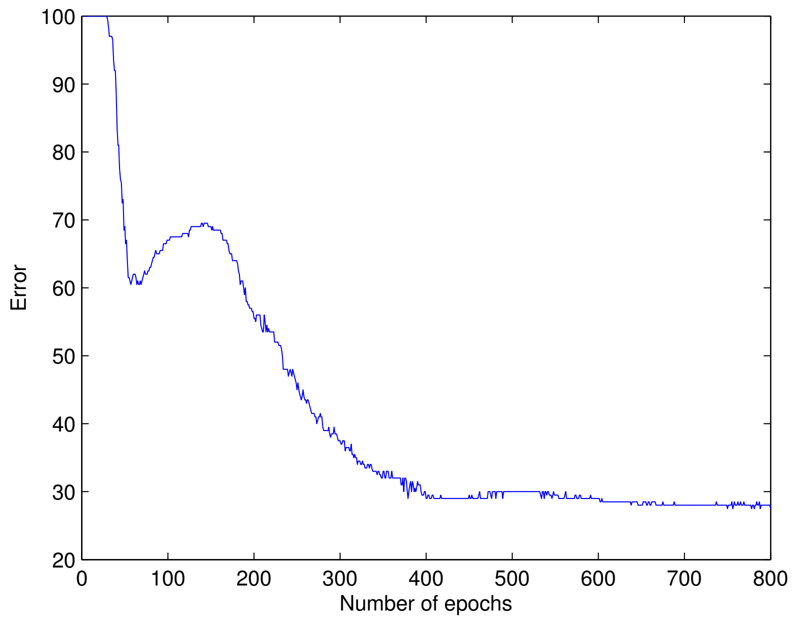

Figure 2: Learning rate 0.01

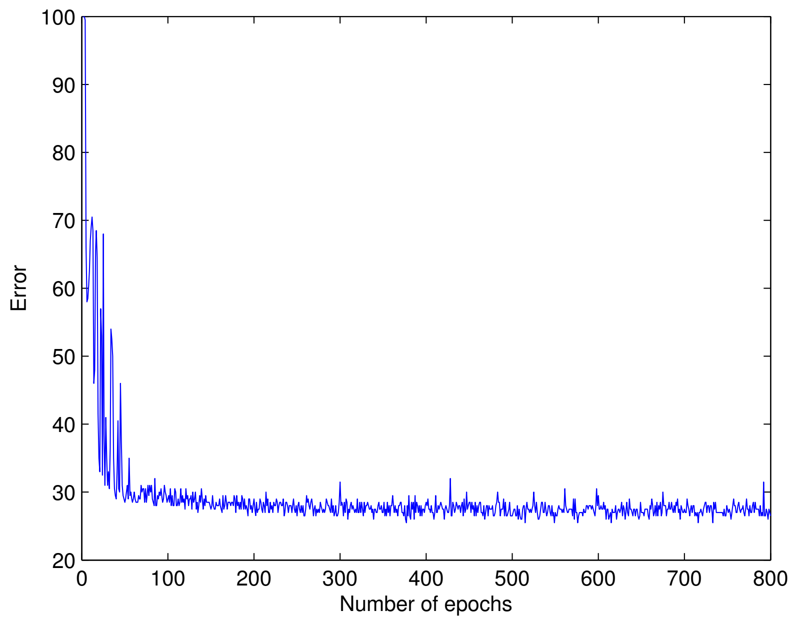

Figure 3: Learning rate 0.1

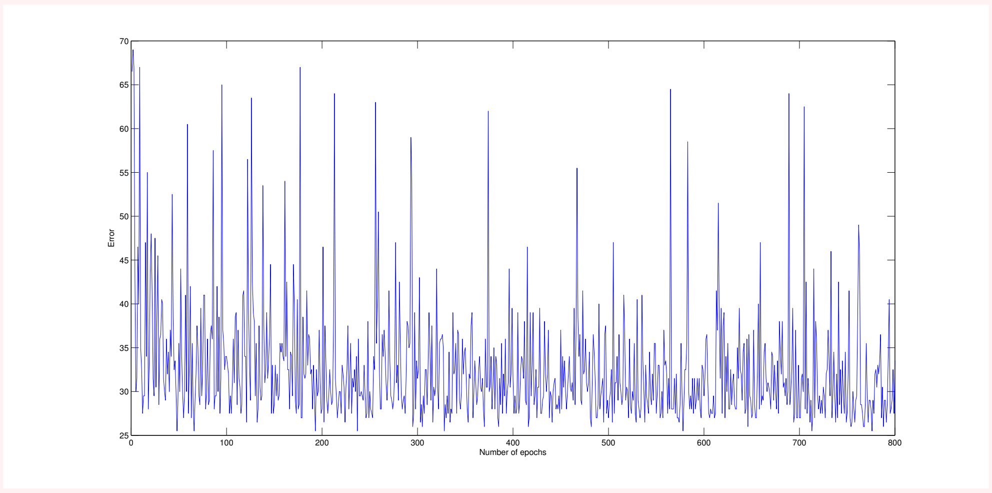

Figure 4: Learning rate 1

Conclusions:
We choose to use the learning rate ‘0.1’ because it converges faster than 0.001 & 0.01 and is more stable than learning rate 1. (See Figure 1-4)

In general, if the learning rate is high, the convergence is faster and it is slower when the learning rate is too low.

Plotting

- See *predict.m*
- See ‘plot_boundary.m’ for data & decision boundary plot.

For 1 hidden neuron, we do not have a decision boundary because our network classifies every input as belonging to one class. So we cannot use the meshgrid and contour to plot.

The plots for the rest are as follows:

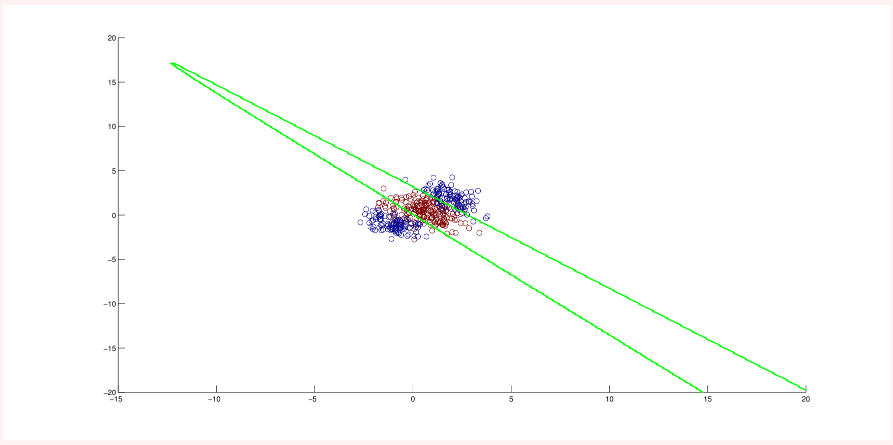

Figure 5: Decision boundary for 2 hidden Neurons

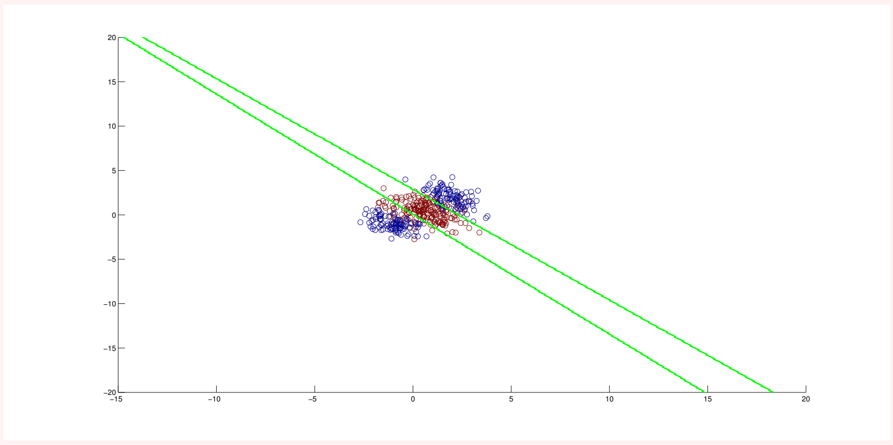

Figure 6: Decision boundary for 3 hidden Neurons

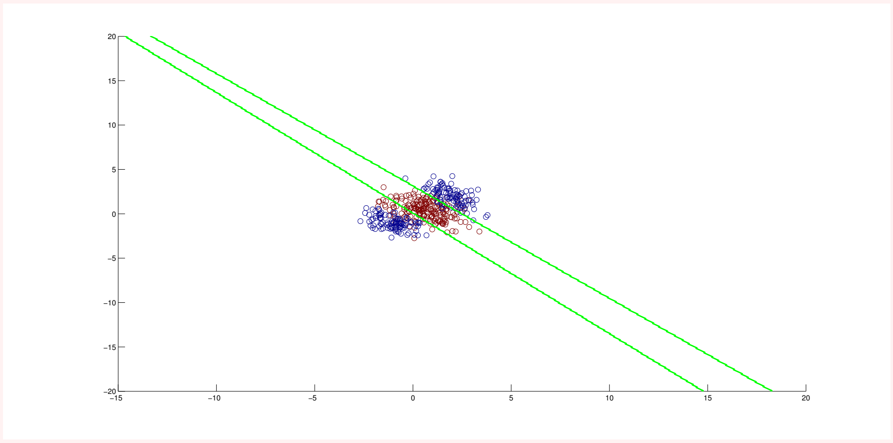

Figure 7: Decision boundary for 4 hidden Neurons

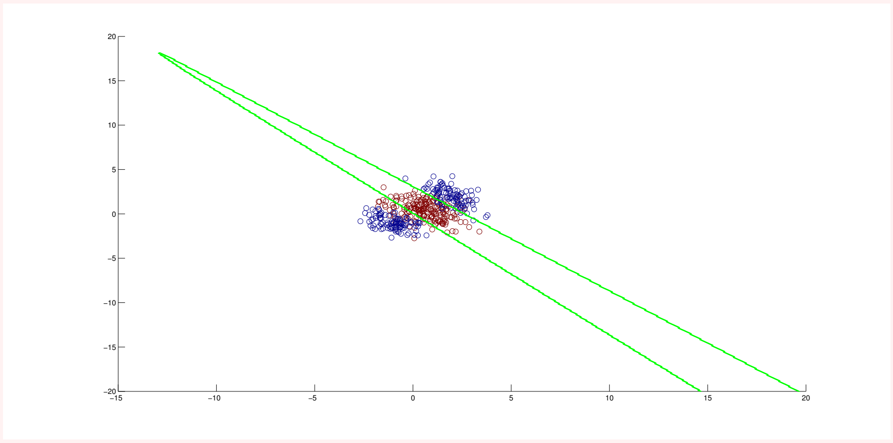

Figure 8: Decision boundary for 5 hidden Neurons

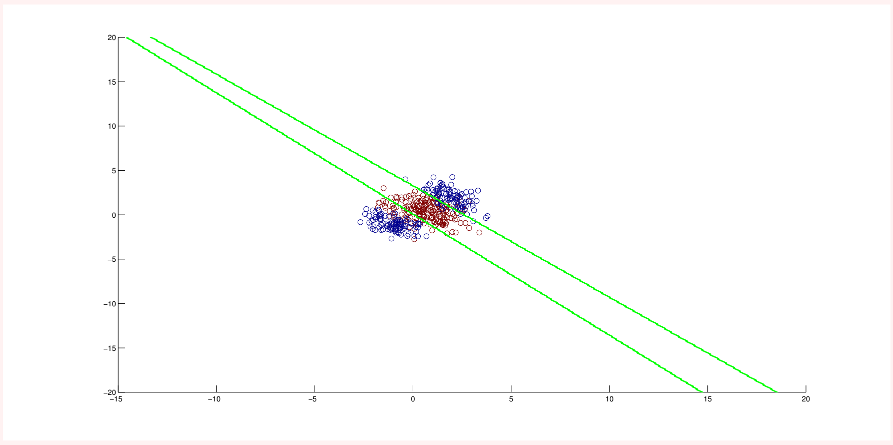

Figure 9: Decision boundary for 6 hidden Neurons