Paul Johnston

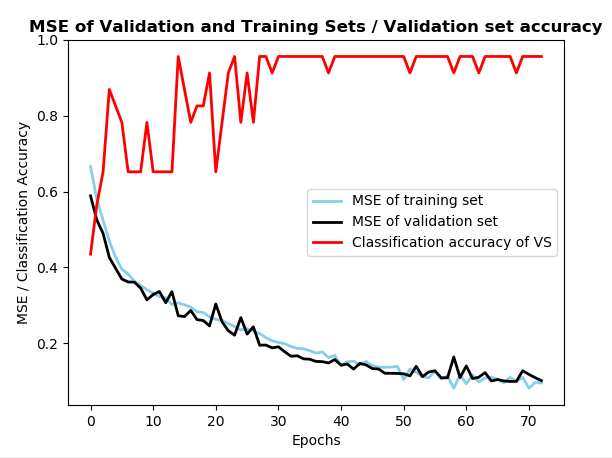
CS 478

MLP with Backpropagation

1. This MLP with Backpropagation (Backprop for short) has been designed to handle an arbitrary number of hidden layers with an arbitrary number of nodes for each layer. Initially, the weights between nodes are randomly set with random values between negative one and one, with a mean of 0. It uses stochastic weight updates meaning it changes the weights after each instance. It uses a non-linear activation function (in this case the sigmoid function) as well as the standard fprime(net) to determine error values for nodes. It separates out a validation set with which to test against at each epoch and determine if further epochs should be explored. It also uses an optional adaptive learning rate which is adding the change of a weight by the previous weights change multiplied by a “momentum” rate which can be set by the user. The stopping criterion is based on how many epochs the user is willing to explore with no decrease of the Mean Squared Error of the validation set.

2. Iris Dataset

The graph below was produced using a learning rate of 0.1 and a momentum term of 0. There was one hidden layer with the number of nodes of the hidden layer being twice the number of nodes for the input layer. The data was split 75/25 into training/testing respectively. A validation set which was 20 % the size of the testing set was put aside for measuring accuracy at the end of each epoch. The final validation set accuracy was 96 percent, the final testing set accuracy was 95 percent. The MSE of the training set was computed for each epoch as well as the MSE for the validation set. Near the end of the epochs, the validation set accuracy was slightly declining while the training set accuracy was slightly increasing. This was expected as the network began to fit the training data more closely, while generalizing less, which is what the validation set measured. After 5 epochs of no improvement of the validation set MSE, the network stopped. Both the MSE’s for the validation and training set are shown alongside the overall accuracy of the validation set as the epochs progressed. The scale of the y axis is consistent for both purposes.



3. Vowel dataset

The baseline accuracy of this task would be about 9 percent with no machine learning involved, just random chance. The input features I chose to use were all but the first three: Train/test, speaker, sex. I didn’t train/test as a useful feature because the manager would split up the data into training and testing portions regardless, the speaker name is not a general feature that will determine the vowel pronunciation, for example all Steve’s don’t have the same pronunciation. The sex feature I didn’t see as useful because again, all males or all females don’t have the same pronunciation among sexes. I kept the other ones because they seemed more a measurement of the actual pronunciation of the vowels. There was one hidden layer with twice as many nodes as the input layer. The data was randomly split 75/25 for training/testing and within that training set, a 80/20 split was made for training/validation. Several learning rates were applied which are shown in the table below along with the corresponding best validation set MSE’s. Each learning rate was tested once.