

Problem assignment 5

Due: Wednesday, February 25, 2003

Problem 1. Logistic regression for Pima dataset

We performed the initial exploratory analysis of the "Pima" dataset in Homework 1. In this homework we are interested in predicting correctly the class label for the signs of diabetes. The dataset (*pima.txt*) and its description (*pima_desc.txt*) can be downloaded from the web. The dataset has been obtained from the UC Irvine machine learning repository *http://www1.ics.uci.edu/~mllearn/MLRepository.html*.

First we try the logistic regression model in combination with gradient methods. Give solutions to the following tasks:

- (a) Write a program that normalizes inputs in the pima dataset (there is no need to normalize outputs) and splits the set to the training and test set, such that 70% of examples are in the training training set and 30 % in the test set. Store the data in *pima_train.txt* and *pima_test.txt* sets. Hint: use functions you wrote in problem set 1 for this purpose.
- (b) Implement a batch-mode gradient procedure, that is, the gradient method in which all data points are considered at the same time. Recall you were asked to write the procedure in problem set 4.
- (c) Implement and submit a program *main1.m* that runs the gradient procedure on the training dataset for 2000 iteration steps (also called epochs). Initialize all weights to 1 at the beginning. Use $2/\sqrt{i}$ learning rate schedule.
- (d) Include graph functions for monitoring the progress of errors in *main1.m* as used in the previous problem sets. Compute mean misclassification error for both the training and testing data at the end. Report your results.
- (e) Experiment with the learning algorithm by changing initial weights, learning schedule, number of epochs. Report training and test misclassification errors. What was the best result you could get?

Problem 2. Logistic regression in Matlab

The logistic regression model and the procedure for training weights can be implemented in Matlab using functions in the Neural network toolbox. File *logistic_NN.m* illustrates how one can build, train and simulate the model. Tasks:

- (a) Familiarize yourself with the program in *logistic_NN.m* and NN toolbox. Run the program for 2000 epochs and calculate the mean misclassification errors for the training and testing data. Report your results. Note that the neural network toolbox lets you to observe the errors on fly. However, also note that it uses LMS errors and not missclassification errors.
- (b) Program *logistic_NN_graphs.m* modifies *logistic_NN.m* and allows you to plot the progress of mean misclassification errors during learning. Report the results obtained by the program and its analysis.
- (c) Experiment with the *logistic_NN_graphs.m* Try to change the parameters of the model, such as the optimization method and the number of epochs. Report the weights with the best mean misclassification rate for the test set and graphs you have found interesting.

Problem 3. Multilayer neural network

The limitation of the logistic regression model is that it uses a linear decision boundary. One way around this is problem is to use non-linear features in combination with a linear model. However, in this case feature function must be fixed and selected in advance. Multilayered neural networks allow us to represent non-linear models by cascading multiple adaptive logistic units. Multilayered neural networks can be built relatively easily with the NN toolbox. File *multi_NN.m* illustrates how one can build, train and simulate a multilayer neural network with 2 hidden units. Tasks:

- (a) Familiarize yourself with the program in *multi_NN.m* and NN toolbox capabilities to model multilayer networks. Run the program for 2000 epochs and calculate the mean misclassification errors for the training and testing data. Report errors and compare them to results obtained for the logistic regression model. Which model is better? Why?
- (b) Run *multi_NN_graphs.m* to obtain the graphs for the misclassification errors, report the graphs and its analysis.

- (c) Experiment with neural networks with 2, 3, 5 and 10 hidden units, while changing other learning parameters, e. g. the optimization method or number of epoch. Analyze and compare the results. Submit the misclassification error graphs.