Learning from Data, Assignment Sheet 2

School of Informatics, University of Edinburgh

Instructor: Amos Storkey

Handed out Friday 9 November 2007.
Submission Deadline: 23:59 Mon 26 November 2007.

Please remember that plagiarism is a university offence. Do not show your work to anyone else. Please also remember that, on any course, you learn as much or more from your peers as you do from your tutors. Please feel free to discuss the general problems with one another (ideally after you have looked at them yourself). But at the end of the day what you write must be yours, and you must understand what you write, and why you didn't write other things. The approach should be one you have chosen to take. If you don't understand it don't write it — it will generally be obvious you don't understand.

The number of marks assigned to each task is given in square brackets. In total, this assignment will contribute 12% to your overall mark for LFD. Please remember that late submissions are not allowed without good reason. Last minute issues (problems of any sort occurring in the last 24 hours) are generally not considered good reason as some contingency should be allowed for. I recommend submitting things 48 hours before the deadline. You can always submit again later if you need to. Presuming a moderate understanding of the course material, this coursework should take *at most* 12 hours, including some time familiarising yourself with MATLAB. The first questions are more practical. The last two involve more thinking. Please use the code given here and in the notes to help. Those with absolutely no MATLAB background may need a little more time getting familiar with MATLAB. Many code examples are given in the assignment. Don't use them blindly. MATLAB will also be needed in the second assignment, so use this one as a means to familiarise yourself.

Please "hand in" your submission electronically, using the `submit` program. Put your answers in plain ascii text (no html etc) in a file named `answers.txt` in a directory named `answers`, along with any code you wrote for the project. Then from a DICE directory that contains answers as a subdirectory type

```
submit msc lfd-5 2 answers
```

if you are an MSc student or

```
submit ai4 lfd-4 2 answers
```

if you are a third or fourth year student. Failure to follow these instructions could result in no marks being given. Note the 2 to distinguish this submission from that for the second

assignment. Typing a 1 instead will result in your submission not being received and zero marks being given. Using a directory with a different name (other than `answers`) could result in your answers being missed and a zero mark. Be warned. Be careful.

I repeat: Typing a 1 instead will result in your submission not being received and zero marks being given. Using a directory with a different name (other than `answers`) could result in your answers being missed and a zero mark. Be warned. Be careful.

The questions ask you to do some MATLAB programming. It is assumed that some of the time of this assignment will involve familiarising yourself with MATLAB. See `http://www.mathworks.com/access/helpdesk/help/techdoc/matlab.html` for some helpful documents, especially the *getting started* section. The routines that you write to accomplish this assignment should be submitted as part of your answer. You are also asked to provide some written answers – these should be written in a plain text file named `answers.txt`. Include your NAME and STUDENT NUMBER at the top of the file, and the LEVEL YOU ARE TAKING THIS COURSE AT (LEVEL 10 or LEVEL 11).

Do not include markup (e.g. html). Keep your file within 80 characters width for ease of printing. Do not include answers in the code. Make it clear what question you are answering at each point. Follow the instructions carefully. You should try to make your code clear and comment it. The code will only be looked at if there are any questions regarding the originality of the answers given, or where an unforseen ambiguity arises. Do not include your answers in the code. The marks will be given on the basis of the written work. The code will not contribute to the marks itself.

Despite the warnings some people failed to submit the first assignment correctly. Remember. Name it `answers.txt`. Put it in directory named `answers`. Make sure it is plain ascii. Don't put any answers anywhere else except in this file. Submit it properly. Submit it in good time.

The marks associated with each question are given. Good insightful answers to questions can serve to increase the baseline marks by at most a further 10 percent up to a maximum of a hundred percent. Verbosity will be penalised.

This assignment is about Neural Networks, and Regularisation and Gaussian Mixture Models.

The data for this assignment is taken from the "Waveform Database" available from the UCI repository. It has been preprocessed into training validation and test sets for you, and is available at

`http://www.inf.ed.ac.uk/teaching/courses/lfd/lfdassignments.html`. Information on the data is available from

`ftp://ftp.ics.uci.edu/pub/machine-learning-databases/waveform`.

Interestingly neural network methods were appalling at the doctored data from assignment one. Typically most networks performed worse than a dumb class probability

predictor. There are good reasons for this. If you are interested, check out the decision boundaries produced by the Gaussian models in assignment one, and consider what a neural network would have to do to represent them. Basically it will involve too many features for too small a dataset. The problems with over-fitting will dominate. Anyhow I digress.

Please download the data. Load into MATLAB using the `load` command. The test data should only be used for evaluation of the methods you develop. The data file consists of 6 MATLAB variables: the training data and training targets, the validation data and validation targets and the test data and test targets. Each row corresponds to a different record (i.e. different data point, different sample etc.). The target variables have only one column, which contains the target label. The data variables have one column for each attribute.

In calculating any covariances (which I don't think you should need to do in this assignment, but I'll leave this in in case), please use the form provided by the MATLAB function `cov` throughout, even though this is not strictly the maximum likelihood estimate (it normalises by $N-1$ not N for number of samples N).

1. [10 Marks] **Rescaling** Why is it good to rescale the data? Why is it better to compute the rescaling on the basis of the training data alone rather than the training and test data? How should the data be rescaled?

2. [30 Marks] Train a suitable neural network with one hidden layer on the rescaled training data. You should use the NETLAB toolbox for neural networks. Type `help netlab` in MATLAB. Describe the form of the neural network you use. Note you will need the 'softmax' activation function. This works in NETLAB by having 3 output units and setting the output activation function to 'softmax'. The output units will then produce probabilities, normalised to one across all three output units. You will need to re-represent the targets using 1-of-m encoding to use them with this network. **A second dataset containing rescaled data and 1-of-m encoded targets is also available from the website** `http://www.inf.ed.ac.uk/teaching/courses/lfd/lfdassignments.html`. Please use this.

Using the validation set assess a good choice of number of hidden layer neurons, random seeds etc. Choose many different runs, but report on about 20 different runs. You will need to initialise your network using `mlp`, and train using `mlptrain`, or better still `netopt` and the scaled conjugate gradient `scg` algorithm. Type `help scg` to get a list of options. I would advise setting OPTIONS(1) to 1. (If you can be bothered you can also see how much slower standard steepest descent methods are, but you don't have to). To repeat the network training with different values, you can use both `rand('state',s)` and `randn('state',n)` to reset MATLAB's random number generator seeds, by choosing some s. Use widely differing s.

You will then have to compute the errors on the validation (and test) data. To do this you need to use `mlpfwd` to get the predictions for the validation and test data. These can be compared with the true values.

There are two possible error measures for comparisons. Method 1: errors - choose

the most likely predicted class from the output probabilities, and give an error rate (a confusion matrix gives more information but makes comparisons harder) for the results. Method 2: predictive log probability - sum up the log of the predicted probabilities of the predicted class of each case over all of the validation (or test) points (`mlperr` will produce the negative of this). This takes account of the uncertainty in the prediction, which the error on the most likely prediction does not. Note that you should compute the "dumb" performance error (just predicting the proportion of each class) for comparison purposes. Using `mlpfwd` will give you the predictions the network is making - if all the predictions are the same the network is doing something dull. You will need more iterations for larger networks. But try various numbers of iterations.

Carefully describe the form of your choice of network, and the exact process you went through in obtaining a good final network (give a table of the settings for each run and the training and validation errors). Be systematic in your procedure, and explain why your system is a good one for exploring the solution space. Justify the decisions you made in this process. What is the performance of the final network of the test data? Comment on the relationship between the performance of the final network on the training, validation and test data. You should use the `NETLAB` toolbox for neural networks. Type `help netlab` in matlab.

3. [10 Marks] Does running the training for as long as possible always result in the best performance on the validation set? If not, why do you think this is?

4. [20 Marks] You can regularise your neural network using the PRIOR term in the `mlp` function. Note the prior term is an inverse variance. Larger values for PRIOR means stronger regularisation.

Using the training data, train a neural network with 20 hidden neurons using various regularisation strengths. Use the same seeds for different regularisation values, but also try many different seeds (local minima may well be an issue in this dataset). Choose the best regularisation strength by the performance on the validation data (choose a larger number of iterations e.g. 200 or more). Again draw a table describing your tests, and your conclusions.

5. [10 Marks] Combine your training and validation sets, and retrain the two best network forms (with and without regularisation) using this combined set. Compare the results of each, and the results based on training on the training data alone. Comment on what you observe.

6. [10 Marks] Presume you are given special permission to use available unlabelled information from additional unlabelled data, combined with all the training data (including the known targets) to help in forming a model to classify the test data. Describe what you could do to use a Gaussian mixture model to achieve this for this data. You do not need to implement this. Nor do you need to go into detail about Gaussian mixture model learning. All that is needed is how you would arrange the data, and what you would do differently from normal Gaussian mixture model training.

Notes on Matlab

- Remember, there are only 100 licences. If you have finished using MATLAB, quit from your session so that others can work.

- On-line documentation about NETLAB can be found at `http://www.dai.ed.ac.uk/dai/computing/software_manuals/netlabhelp/index.htm`, at the netlab website: `http://www.ncrg.aston.ac.uk/netlab/index.php` and in the text book Netlab: Algorithms for Pattern Recognition which is available in the library. You should run the netlab demos and refer to the netlab demo code (e.g. `type demmlp1`, `type demmlp2`) to help you. Allow yourself some time to familiarise yourself with netlab. `help netlab` gives a list of all the netlab functions.

- You can find out more about NETLAB functions (and many MATLAB functions) by typing `help` followed by the function name. Also, you can find the `.m` file corresponding to a given function using the `which` command, for example

  ```
  >> which mlp
  /usr/local/lib/matlab/toolbox/local/netlab/mlp.m
  ```

  You can then examine the code (but this should not be necessary for the assignment). Note you should ensure the netlab toolbox is on the path using `addpath`.

- The command `clf` is very useful; it clears the current figure. This is useful as the functions that you have have `hold on` set, which means that successive plots are plotted on top of each other. If this has become confusing, `clf` clears the figure so you can start again. `clear` clears your workspace of variables (check that the command `who` returns nothing).

- If you wish to find out how much cputime a function is taking, use the online help `help cputime`.

- The functions `mlp`, `netopt` and `mlpfwd`. These are the main NETLAB functions used in this assignment. `mlp` sets up a MLP network. In the call

  ```
  net = mlp(nin, nhidden, nout, transfunc);
  ```

  where transfunc determines the type of transfer function the output unit has.

  `netopt` is a wrapper function that calls the various optimization routines, `scg.m` in this assignment.

  `mlpfwd` forward propagates inputs through the network, to produce the corresponding outputs.