Machine Learning, IUB Fall 2003, Final Exam

December 11, 2003, 9:30

Note: a maximum of 100 points is accredited for this exam.

All problems are of about the same difficulty. The points reflect the required amount of work to be expected, not difficulty. Good success!

Overview.

1 **Problem 1:** *n* times 6 points; *n* is up to you. Topic: General discussion of a very difficult learning task.

2 **Problem 2:** 33 points in total, many subtasks. Topic: explanation of Bayesian approach to digit classification learning.

3 **Problem 3:** 25 points in total, two subtasks. Topic: compute weights of an optimal linear filter.

4 **Problem 4:** 20 points in total, two subtasks. Topic: informal analysis of LMS for adaptive system identification.

5 **Problem 5:** 20 points in total, two subtasks. Topic: informal analysis of overfitting prevention in an MLP.

Problem 1 (*n* times 6 points; *n* is up to you). Assume you want to predict the Dow Jones stock market index. What you have in your hands is the chart of this index up to NOW, that is, a real-valued time series $D_1(t_i^1)$, $i = 1, ..., N^1$, where the t_i^1 are all the time points when the Dow Jones index was recorded. In addition, you have many other sources of information, for instance other stock market indices, other economical time series like currency exchange rates, inflation rates, unemployment rates of various countries, -- you name it, for any practical purpose there is an unlimited supply of economically relevant time series. Let's name them $D_n(t_i^n)$, where $n = 1, 2, ...$ is an index distinguishing the various time series, $t_i^n = 1, ..., N^n$ are the time points where the *n*-th time series is measured. These are the training data for learning a predictor for D_1 beyond time $t_{N^1}^1$ = NOW. Predicting stock market indices is a difficult task, and many people have tried it with little success (it's much safer to get rich by selling stock market prediction algorithms than by actually using them!). Your task: give reasons why this is a difficult task in the perspective of machine learning and suggest approaches to overcome the difficulties.

Delivery format: Present each difficulty you can think of (i) by a short description of the difficulty (target size: 2 lines of text), (ii) by a sketch of a possible counter-measure to overcome or at least mitigate the difficulty (target size: 50 to 100 words). Grading: maximally 6 points for each difficulty you treat.

Problem 2 (33 points in total). Consider the ten digits classification task. Specifically, assume that your training data are pixel images of handwritten digits, *N* instances per digit, and you want to use these to train a classifier that on input of a new (test) pixel image, returns a decision which digit was presented.

a. (5 points) Compare a situation A where you have many training patterns (let's say, N = 100,000) and a situation B where you have very few (e.g., $N = 30$). Explain what general approach to learning a classifier should be taken in A vs. B, and why. Deliverable: about 4 lines of text.

In the remainder of this problem, assume that you have chosen to take a Bayesian approach to training a classifier. Specifically, assume that each digit image i is represented through a vector of 10 feature values $\mathbf{x}^i = (x_1^i, ..., x_{20}^i)$, where the features $x_1, ..., x_{20}$ are uncorrelated and each of them is distributed normally within each digit class. That is, each of the ten digit classes is characterized through a 20-dimensional normal distribution with 40 parameters $\theta^c = (\mu^c_1, ..., \mu^c_{20}, \sigma^c_1, ..., \sigma^c_{20})$, where μ^c_k and σ^c_k are the mean and the standard deviation of the k-th feature in class c ($c = 1, ..., 10$ and $k = 1, ..., 20$). The starting point and central part of Bayesian classification learning is to estimate the θ^c from the training data, incorporating prior knowledge. Answer the following questions:

b. The starting point for estimating the θ^c is Bayes formula $p(\theta^c | D) = p(D|\theta^c)\, p(\theta^c)$ / NT, where NT is a normalization term that we will ignore here [for those who are interested: NT = $\int_{\theta^c} p(D\,|\,\theta^c) p(\theta^c) d\theta^c$]
   i. (4 points) What, concretely, is D in this case of estimating θ^c?
   ii. (4 points) What is $p(\theta^c)$ [name it and explain in 2 lines]? What is the dimension of the distribution $p(\theta^c)$, that is, what is the dimension of the measure space?
   iii. (4 points) What is $p(D|\theta^c)$ [explain in 2 lines]?
   iv. (4 points) What is $p(\theta^c | D)$ [name it and explain in 2 lines]? What is the dimension of this distribution?

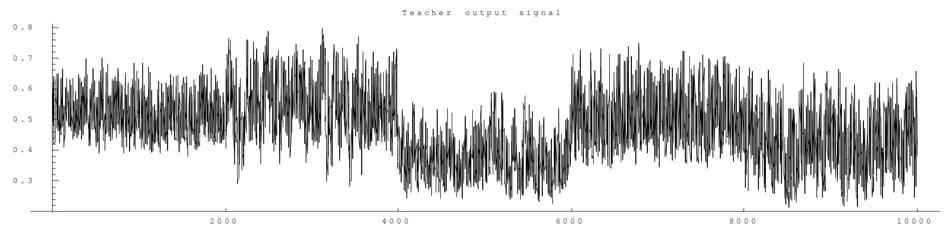
c. (6 points) When working with the formula $p(\theta^c | D) = p(D|\theta^c)\, p(\theta^c)$ / NT, you have to concretely specify $p(\theta^c)$. Because this is the point where your background information enters, you are free to use any distribution you deem appropriate. Suggest one approach. Hint: one natural way to enter background information about the digit class c is to think of a prototype exemplar digit representative of this class, and derive $p(\theta^c)$ from that. Deliverable: about 6 lines of text.

d. (6 points) Assume you have successfully found estimates $\theta^{c\text{ PME}}$ for θ^c, thus you have learnt to describe the feature distributions within each digit class. Now you are confronted with a new test pattern $\mathbf{x}$. You know that in the population of digits where your test pattern comes from the ten digits occur with probabilities $P_1, ..., P_{10}$. Describe how you can compute P(test pattern is in class i | $\mathbf{x}$) using the information of the $P_1, ..., P_{10}$ and the estimates $\theta^{c\text{ PME}}$. Deliverables: the version of Bayes formula that you need here and an expression to compute P($\mathbf{x}$ | test pattern is in class c).

Problem 3 (25 points). Consider a linear system with the update equation $x(n) = 2u(n) + 3u(n-1) - u(n-2) + v(n)$, where $u(n)$ is a white noise input signal with variance $\sigma_x^2 = 1.0$ and $v(n)$ is an additive white noise with $\sigma_v^2 = 0.1$ which is not correlated to any $u(n-k)$, where $k = 0, 1, 2, ...$.

a. (20 points) What are the weights w_1 and w_2 of an optimal transversal filter of the form $y(n) = w_1\, u(n) + w_2\, u(n-1)$? Derive your answer either by a direct computation, using the Wiener-Hopf equation, or by guessing the answer and justifying it by invoking the principle of orthogonality.
b. (5 points) What is the residual error $\xi_{\min}$?

Problem 4 (20 points). An unknown, discrete-time, input-output system with one-dimensional output $d(n)$ is driven by a one-dimensional, stationary white noise input signal $u(n)$. The system is known to have a very long memory, that is, the current output depends on very many previous inputs. However, the contributions of inputs that lie distant in the past get asymptotically small. In addition, it is known that after every 2000 updates, the system function f changes. (This is not an uncommon situation: for instance, the system in question might be some combustion engine on a test stand; its dynamic behavior in different operating conditions is systematically explored by changing operating parameters in regular intervals. In such a case, the input $u(n)$ might be the setting of a carburetor valve, and $d(n)$ might be measured torque output.) The figure below shows a typical plot of the observed system output.



Your task: describe a learning setup where the LMS algorithm is used to adaptively identify the system online. Specifically, do the following:

a. (5 points) Sketch a block diagram of how the adaptive LMS filter is set up for this task. There should be arrows for the signals $u(n)$, $d(n)$, for the filter output $y(n)$ and for the error $\varepsilon(n)$. The filter can be drawn by a simple box, you don't have to represent the tap delay structure.
b. (15 points) Consider three cases, where the filter is trained with ESN on the experiment shown in the figure above. The filter is parametrized in three different fashions:
   Case A: long filter (100 tap weights), small learning rate,
   Case B: short filter (10 tap weights), small learning rate,
   Case C: short filter (10 tap weights), large learning rate (but still within stability region).
   Deliverable 1: Draw a schematic plot with time $n = 1, ..., 10{,}000$ on the x-axis and the square error $\varepsilon^2(n)$ on the y-axis. Use a logarithmic scale for the y-axis. Assume that the square error is reasonably smoothed before plotting. Assume that the largest squared error you see over all three experiments is 1.0 and the smallest is 1e-4. Draw the square errors for all three experiments into a single plot. Your plot should reveal the principal differences that can be expected from the three different filter parametrizations. Deliverable 2: Explain in words the expected differences between cases A vs. B, B vs. C, A vs. C. Target size: 3 lines of text for each pairing.

Problem 5 (20 points) Here is a quote from the neural networks FAQ at http://www.faqs.org/faqs/ai-faq/neural-nets/:

```
Neural Network practitioners often use nets with many times as many
parameters as training cases. E.g., Nelson and Illingworth (1991, p. 165)
discuss training a network with 16,219 parameters with only 50 training
cases! The method used is called "early stopping" or "stopped training" and
proceeds as follows:

1. Divide the available data into training and validation sets.
2. Use a large number of hidden units.
3. Use very small random initial values.
4. Use a slow learning rate.
5. Compute the validation error rate periodically during training.
6. Stop training when the validation error rate "starts to go up".
```

Early stopping is one "quick and dirty" but simple and fast approach to tackle the overfitting problem. It uses only a single, possibly oversized network, whose capacity is not fully exploited. A more cautionary, more standard method to prevent overfitting would be to train networks of various sizes, check their generalization abilities with some cross-validation scheme, and choose the network size (and network) that gives best generalization. Your task: compare the two approaches, by answering the following questions:

a. (4 points) How do the overall computational costs of the two approaches compare? List what computations have to be done in each case and compare.
b. (16 points) Early stopping is often suspected to give poorer results (larger risk) than a careful selection of a properly sized network, which is then trained to minimal training error. Explain why this might happen. Illustrate your explanation by drawing hypothetical network output graphs into the diagram below, one graph coming from a carefully sized, fully trained network and one graph coming from an early-stopped large network. Assume that both nets have been trained on the training samples indicated in the figure.

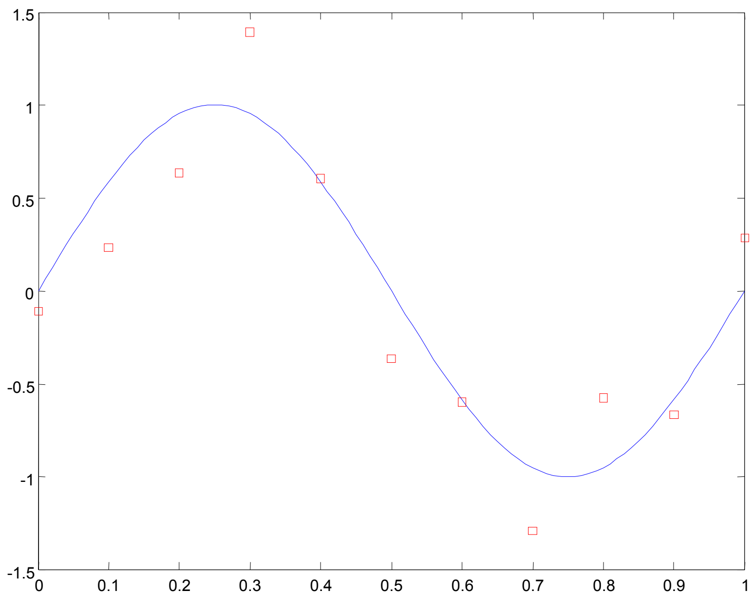


Figure: a nonlinear function (line) and a noisy training sample (squares).