Lehrstuhl für Informatik VI
Rheinisch–Westfälische Technische Hochschule Aachen
Prof. Dr.–Ing. H. Ney

Wednesday, November 30th, 2009
S. Wiesler & M. Ratajczak

7. Exercise Sheet **Pattern Recognition and Neural Networks**

The solutions to the problems indicated with (*...) may be submitted until **Wednesday, December 9th, 2009**, either in the secretary office of the Lehrstuhl für Informatik VI or at the latest before the exercise lesson on the same day. Problems marked with (**...), i.e. two asterisks, are optional.
Please note: for the submission of programming problems we require that you use Matlab/Octave, or C++. Please submit your programs as printout *as well as* via email to <wiesler, ratajczak@informatik.rwth-aachen.de>.

1. ***Maximum Likelihood* Estimation of Continuous Probability Distributions**
   Assume training data is given by a sequence of N observation $x_1, x_2, \ldots, x_N$ with $x_n \in \mathbb{R}$. Derive the *Maximum Likelihood* estimation formula for the following continuous probability distributions $p(x|\,\vartheta)$, i.e. for the parameter ϑ or μ, ν, respectively:

   (a) $$p(x|\,\vartheta) \;=\; \begin{cases} \vartheta e^{-\vartheta x}, & x \geq 0 \\ 0, & \text{otherwise.} \end{cases}$$

   (b) $$p(x|\,\vartheta) \;=\; \begin{cases} 2\vartheta x e^{-\vartheta x^2}, & x \geq 0 \\ 0, & \text{otherwise.} \end{cases}$$ (* 4P)

   (c) Assume an equal distribution with given end point ϑ:
   $$p(x|\,\vartheta) \;=\; \begin{cases} \frac{1}{\vartheta}, & 0 \leq x \leq \vartheta \\ 0, & \text{otherwise.} \end{cases}$$
   Show that the *Maximum Likelihood* estimator for ϑ is the maximal value $x_{\max} = \max_{n=1}^{N} x_n$. (* 3P)

   (d) Derive the *Maximum Likelihood* estimator for the parameters μ and ν of a *Laplace* distribution $p(x|\mu,\nu)$ with
   $$p(x|\,\mu,\nu) \;=\; \frac{1}{2\nu} \exp\left(-\frac{|x-\mu|}{\nu}\right)$$

2. ***Maximum Likelihood* Training of an Image Recognition System**
   A basic approach to image recognition is to use the intensity matrix of an image directly as observation "vector" for the corresponding image recognition system. Using such an approach, a set of preprocessing steps is needed to normalize the data with respect to, e.g., dynamic range of intensity values, position, size, and orientation of the object represented by an image, which is to be recognized. Now assume that these normalization steps are already done, i.e. each image $n = 1, \ldots, N$ of our data base is given by its class k_n and its normalized observation vector $x_n = x_{n,1}, \ldots, x_{n,D}$ of fixed dimension D with $x_{n,d} \in \mathbb{R} \quad \forall\, n = 1, \ldots, N$ and $d = 1, \ldots, D$. Now assume a *Gaussian* classification model, i.e. the class conditional probability of a single observation x of an image of class k is given by:
   $$p(x|k) = \mathcal{N}(x|\mu_k, \Sigma_k)$$
   with (class specific) mean vectors μ_k and covariance matrices Σ_k, i.e. the parameters of the multivariate *Gauss* distributions.

See second page on reverse side!

On our server wasserstoff.informatik.rwth-aachen.de (anonymous ftp) you find data for the so called *US Postal* (USPS) digit recognition task in the directory pub/PatRec/usps/. The corpus consists of handwritten digits extracted from postal codes on letters. The digits are represented by a matrix of 16×16 grey values in the range between 0 and 1000. The data has been transformed into ASCII format and has been formatted such that you will be able to get an idea of the data by displaying the data a font with fixed width. The original corpus has been preprocessed and divided into a training and a testing part, usps.train and usps.test. In usps.README you find a description of the formats of the training and testing data files as well as the parameter files to be produced. In the same place you will also find a solution framework.

(a) Assume the training data is given by a set of images $n = 1, \ldots, N$ which are represented by their classes k_n and the observation vectors x_n with $x_n = x_{n,1}, \ldots, x_{n,D}$ of fixed dimension D with $x_{n,d} \in \mathbb{R} \quad \forall\, n = 1, \ldots, N$ and $d = 1, \ldots, D$. (* 6P)
In the lecture notes you will find results for *Maximum Likelihood* estimation of the parameters of a multivariate *Gaussian* probability distribution with class specific as well as pooled covariance matrices. Give derivations of the *Maximum Likelihood* estimation of multivariate *Gaussian* probability distributions with:

i. full class specific covariance matrix Σ_k

ii. diagonal class specific covariance matrix $\Sigma_{k,dd'} = \begin{cases} \sigma_{kd}^2 & \text{for} \quad d = d' \\ 0 & \text{otherwise} \end{cases}$

iii. pooled covariance matrix Σ

iv. pooled diagonal covariance matrix $\Sigma_{dd'} = \begin{cases} \sigma_{d}^2 & \text{for} \quad d = d' \\ 0 & \text{otherwise} \end{cases}$

Hints: When computing derivatives with respect to vectors and matrices try to do this component-wise before returning to a matrix or vector representation. Further useful formulae:

$$\frac{\partial \det A}{\partial A_{ij}} = (A^{-1})_{ij} \cdot \det A \qquad \text{and} \qquad \frac{\partial (A^{-1})_{ij}}{\partial \alpha} = -\left(A^{-1} \cdot \frac{\partial A}{\partial \alpha} \cdot A^{-1}\right)_{ij}$$

In case of derivatives with respect to (full) covariance matrices which are to be set to zero, you can equally well use the derivative with respect to the inverse of the matrix. Doing this you should consider the following for the determinant of the inverse of a matrix A: $\det A^{-1} = \frac{1}{\det A}$.

(b) Implement a *Maximum Likelihood* estimator for the cases in **??**. The corresponding output parameter file should conform with the format given in usps.README. (** 4P)

(c) Using your program, or Netlab, estimate the parameters for all cases using the training data usps.train. Explain, why the parameter sets with full and diagonal (non-pooled) covariance matrices cannot be used for recognition without further model assumptions. (* 4P)

(d) Reshape the *Gaussian Bayes* classifier from Exercise 4 to conform with the format of the parameter file produced in **??** and the testing data file given here. You may also start with the solution presented in the exercise lessons, which is to be found on our ftp-server in pub/PatRec/exercise5problem2/solution/. (** 1P)

(e) Using your program, or Netlab, calculate the error rates and confusion matrices using pooled diagonal and pooled covariance matrices. (* 4P)

(f) As stated in **??**, the full and diagonal covariance matrices cannot be used for recognition without tying these parameters over all classes (pooling). The results for the given problem can now be improved by using the following covariance model: (* 4P)

$$\tilde{\Sigma}_k = \lambda \Sigma + (1 - \lambda) \Sigma_k$$

with the full and pooled covariance matrices as estimated in **??** and a real parameter $\lambda \in \,]0, 1]$. Calculate the error rates for the following values of $\lambda = 1, 0.5, 0.1, 0.01, 10^{-3}, 10^{-4}, 10^{-5}, 10^{-6}$. Plot the resulting error rates in % using a logarithmic axis for λ. Interpret your results.