Assignment 2: Basic Learning Algorithms

Christian Igel and Kim Steenstrup Pedersen

February 2011

This assignment is based on parts of chapters 1-4 from C. M. Bishop *Pattern Recognition and Machine Learning*. The goal of this assignment is to get familiar with basic supervised learning methods.

You have to pass this and the following mandatory assignments in order to be eligible for the exam of this course. There are in total 3 mandatory pass/fail assignments on this course, which can be solved individually or in groups of no more than 3 participants. The course will end with a larger written exam assignment which must be solved individually and is graded (7-point scale).

The deadline for this assignment is Tuesday 8/3 2011. You must submit your solution electronically via the Absalon home page. Go to the assignments list and choose this assignment and upload your solution prior to the deadline. If you choose to work in groups on this assignment you should only upload one solution, but remember to include the names of all participants both in the solution as well as in Absalon when you submit the solution. If you do not pass the assignment, having made a serious attempt, you may get a second chance of submitting a new solution.

A solution consist of:

- Your solution source code (Matlab / R / Python scripts / C or C++ / Java code) with comments about the major steps involved in each Question (see below).
- Your code should also include a README text file describing how to compile and run your program, as well as list of all relevant libraries needed for compiling or using your code. If we cannot make your code run we will consider your submission incomplete and you may be asked to resubmit.

- A PDF file with notes detailing your answers to the non-programming questions, which may include graphs and tables if needed (**Max 10 pages** text including figures and tables). Do NOT include your source code in this PDF file.

1 Regression

In the regression part of the case we will study regression with linear models (as explained in Bishop Sec. 3.1) and we will experiment with different basis functions. The data set we will consider is a real-world data set which contains a set of estimates of the percentage of body fat determined by underwater weighing and various body circumference measurements from 252 men. We will try to make predictions of the percentage of body fat based on circumference measurements using linear regression models.

You can load the data set with the function `readbodyfat` (available for Matlab and R). Since the data set seems to be ordered somewhat with respect to the age of the test subjects, we should pick the members of the training and test sets at random. After loading the dataset the function performs a random split of the data set into a training and test set (in roughly 80% training and 20% test).

You find a detailed explanation of the variables in the dataset in the file `bodyfat.txt`, but the short description is that we would like to predict percentage body fat given in the 2nd column from a subset of the variables given in columns 4 to 15. We will not use the values found in column 1 (column 2 is actually a function of column 1) and column 3 (the age of the subject). We will think of column 2 as what is referred to in the book as the target variable t and a selection of columns 4 to 15 as the observations $\mathbf{x}$. We will in the following consider the following two selections of observation variables:

Selection 1: Let $\mathbf{x}$ consist of the data in columns 4, 7, 8, and 9, hence the dimensionality of this subset is $D = 4$.

Selection 2: Let $\mathbf{x}$ consist of the data in column 8, hence the dimensionality of this subset is $D = 1$.

1.1 Maximum likelihood solution

We will start by trying to model the data set with linear regression as defined in eq. (3.1) in the book and learn the parameters using the maximum likelihood approach. That is, we will use the linear model

$$y(\mathbf{x}, \mathbf{w}) = w_0 + w_1 x_1 + \ldots + w_D x_D \ .$$

Deliverables: Implement this model for variable selections 1 and 2 by constructing their corresponding design matrices as defined in eq. (3.16). Train each of these models on the training set by finding the maximum likelihood (ML) estimate by using eq. (3.15) (hint: In Matlab you can compute the pseudo inverse of the design matrix with the function `pinv` and in R by using `ginv` or `pseudoinverse`). Apply each model to the test set using eq. (3.3) with the ML parameter estimate and compute and report the root mean square (RMS) error

$$\mathrm{RMS} = \sqrt{\frac{1}{N}\sum_{n=1}^{N}(t_n - y(\mathbf{x}_n, \mathbf{w}))^2} \ .$$

Which of the two models seem to provide the best prediction?

1.2 Maximum a posteriori solution

Next let us try to learn the two models using Bayesian learning and maximum a posteriori (MAP) estimation. Lets fix the prior distribution on the parameters to the zero mean isotropic Gaussian as given in eq. (3.52). Set the noise precision parameter to $\beta = 1$. We may then obtain the posterior distribution as given in eq. (3.49), by computing the estimates of the mean $\mathbf{m}_N$ and covariance $\mathbf{S}_N$ using eq. (3.53) and (3.54). The posterior mean is the MAP estimate.

Deliverables: Using the MAP estimate and eq. (3.3) apply each model to the test set and compute and plot the root mean square (RMS) error for different values of the prior precision parameter α. Which of the two models seem to provide the best prediction? How does the results compare with the maximum likelihood results? For what value of the prior precision parameter α does the RMS error go below the RMS for the maximum likelihood solution from Question 1.1?

2 Linear Discriminant Analysis

Download the file archive `knollData.tgz` from the course homepage.

It contains samples from three different artificial binary classification problem. These problems belong to a family of problems we refer to as KNOLL. The files `knollA-train.dt` and `knollA-test.dt` contain training and test data, respectively, of the first problem KNOLLA, the files for the other two problems KNOLLB and KNOLLC are named accordingly. In these problems, two real numbers are mapped to two classes indexed by -1 and 1. Patterns from class one are drawn from a single Gaussian distribution, patterns from the other class from either a single Gaussian or a mixture of two Gaussians depending on the problem.

Visualize the training data sets in three 2D plots.

First, we consider building a linear model for classification. Apply linear discriminant analysis (LDA) to the training data sets and report the accuracies of the classifier on the training as well as on the test sets. Explain the results; discuss similarities and differences in performance on the three data sets and if you think a non-linear method could do better.

It is highly recommended that you implement the LDA algorithm by yourself. However, it is acceptable to use a software tool.

Deliverables: three plots of the training data; implementation of LDA or description of tool used for LDA; results of LDA applied to the three problems, short discussion of the results

3 Nearest Neighbor Classification

Linear discriminant analysis is a linear technique using a parametric model. Now we apply a non-linear, non-parametric method to the same data, namely nearest neighbor classification.

3.1 Nearest Neighbor Classification with Euclidean Metric

Implement a k-nearest neighbor classifier (k-NN). Train it for all three KNOLL problems using the training data sets and report its accuracy on the correspond test sets for $k = 1, 3, 5, \dots, 9$.

Explain the results; discuss similarities and differences in performance on the three data sets and compare the results to those achieved by LDA.

Deliverables: source code of your nearest neighbor classifier; results of k-NN for all three problems using $k = 1, 3, 5, \ldots, 9$, short discussion of the results

3.2 Changing the Metric

The function

$$d(\boldsymbol{x}, \boldsymbol{z}) = \|\boldsymbol{M}\boldsymbol{x} - \boldsymbol{M}\boldsymbol{z}\| \tag{1}$$

with

$$\boldsymbol{M} = \begin{pmatrix} 100 & 0 \\ 0 & 1 \end{pmatrix} \tag{2}$$

and $\|\cdot\|$ denoting the standard L_2-norm is a metric on $\mathbb{R}^2$.

Proof that d is a metric by showing that the opposite cannot be true because $\|\cdot\|$ is a metric.

Use d as a metric in you k-NN classifier and apply it to KNOLLC. Explain the results.

Deliverables: proof that d is a metric; results for the k-NN classifier using the non-standard metric d applied to the KNOLLC data and a short discussion of the results

3.3 High-dimensional Data

When we think about elements of the $\mathbb{R}^n$ with large $n \gg 4$, our intuition rooted in visualizing $\mathbb{R}^2$ or $\mathbb{R}^3$ can be misleading. Because a geometric view on data is extremely helpful in machine learning, it is important to think about differences in high and low dimensional spaces.

In this assignment, we therefore study phenomena related to random vectors. Generate real-valued random vectors of *dimensionality* n, where each component is drawn independently from the same distribution with zero mean and unit variance (e.g., a standard normal distribution). Compute the length (L_2-norm) of the vectors and the Euclidean distances from each other for increasing n. What do you observe? What happens for $n \to \infty$? What does this imply for nearest neighbor classification?

Deliverables: discussion of average length and distance of random vectors in high dimensional feature spaces