Machine Learning 2016/2017, Assignment 6

Yevgeny Seldin, Christian Igel
Department of Computer Science, University of Copenhagen

The deadline for this assignment is **12:00 pm (noon, not midnight) 09/01/2017**. You must submit your individual solution electronically via the Absalon home page.

A solution consists of:

- A PDF file with detailed answers to the questions, which may include graphs and tables if needed. Do *not* include your source code in this PDF file.
- Your solution source code (Matlab / R / Python scripts or C / C++ / Java code) with comments about the major steps involved in each question (see below). Source code must be submitted in the original file format, not as PDF.
- Your code should be structured such that there is one main file that we can run to reproduce all the results presented in your report. This main file can, if you like, call other files with functions, classes, etc.
- 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.

1 Kernel-induced metric

Given a kernel k on input space $\mathcal{X}$ defining RKHS $\mathcal{H}$. Let $\Phi : \mathcal{X} \to \mathcal{H}$ denote the corresponding feature map (think of $\Phi(x) = k(x, .)$). Let $x, z \in \mathcal{X}$. Show that the distance of $\Phi(x)$ and $\Phi(z)$ in $\mathcal{H}$ is given by

$$\|\Phi(x) - \Phi(z)\| = \sqrt{k(x,x) - 2k(x,z) + k(z,z)}$$

(if distance is measured by the canonical metric induced by k).

2 SVM in practice

In this part of the assignment, you should get familiar with support vector machines (SVMs). You need an SVM implementation, and you are welcome to use existing SVM software.

We recommend using LIBSVM [1], which can be downloaded from `http://www.csie.ntu.edu.tw/~cjlin/libsvm`. Interfaces to LIBSVM for many programming languages exist, including Matlab and Python. For R, package such as E1071 (recommended) and KERNLAB are available. If you are comfortable with C++, you are encouraged to use the SVM implementation within the SHARK machine learning library [3]. Get the latest snapshot from `http://image.diku.dk/shark`.[1]

For this exercise, use Gaussian kernels of the form

$$k(\boldsymbol{x}, \boldsymbol{z}) = \exp(-\gamma \|\boldsymbol{x} - \boldsymbol{z}\|^2) \ . \tag{1}$$

Here $\gamma > 0$ is a bandwidth parameter that has to be chosen in the model selection process. Note that instead of γ often the parameter $\sigma = \sqrt{1/(2\gamma)}$ is considered.

Application: Diagnosing Parkinson's disease voice signals

We consider the medical application of diagnosing Parkinson's disease from a person's voice. We consider the data from [4], which can be obtained from the well-known UCI benchmark repository [2].

The data were collected from 31 people, 23 suffering from Parkinson's disease. Several voice recordings of these people were processed. Each line in the data files corresponds to one recording. The first 22 columns are features derived from the recording, including minimum, average and maximum vocal fundamental frequency, several measures of variation in fundamental frequency, several measures of variation in amplitude, two measures of ratio of noise to tonal components in the voice status, two nonlinear dynamical complexity measures, a measure called signal fractal scaling exponent, as well as nonlinear measures of fundamental frequency variation [4, 2]. The last column is the target label indicating whether the subject is healthy (0) or suffers from Parkinson's disease (1). The data were split in to a training and test set, `parkinsonsTrainStatML.dt` and `parkinsonsTestStatML.dt`, respectively.

[1] Carefully study what the SVM implementation you use is doing under the hood. Some implementations consider C/ℓ instead of C in the SVM objective function, where C denotes the regularization parameter. The SVM from the Matlab Bioinformatics Toolbox may by default use different regularization parameters depending on the class and the class frequency.

2.1 Data normalization

As seen in a previous assignment, data normalization is an important preprocessing step. A basic normalization is to generate mean free, unit variance input data. You may reuse the code from your previous assignments.

Consider the training data in `parkinsonsTrainStatML.dt`. Compute the mean and the variance of every input feature (i.e., of every component of the input vector). Find the affine linear mapping $f_{\text{norm}} : \mathbb{R}^{22} \to \mathbb{R}^{22}$ that transforms the training data such that the mean and the variance of every feature in the transformed data are 0 and 1, respectively (verify by computing these values).

Use the function f_{norm} to also encode the test data. Compute the mean and the variance of every feature in the transformed test data.

The normalization is part of the model building process. Thus, you may only use the training data for determining f_{norm} (always remember that you are supposed to not know the test data).

Deliverables: Mean and variance of the training data; mean and variance of the transformed test data

2.2 Model selection using grid-search

The performance of your SVM classifier depends on the choice of the regularization parameter C and the kernel parameters (here γ). Adapting these *hyperparameters* is referred to as SVM *model selection*, as already discussed in a previous assignment.

Use grid-search to determine appropriate SVM hyperparameters γ and C. Look at all combinations of $C \in \{c_1, c_2, \ldots, c_7\}$ and $\gamma \in \{g_1, \ldots, g_7\}$, where you have to choose proper grid points for yourself. Consider values around $C = 10$ and $\gamma = 0.1$ of different orders of magnitude. It is common to vary the values on a logarithmic scale (e.g., $0.001, 0.01, 0.1, 1, 10, 100$). For each pair, estimate the performance of the SVM using 5-fold cross validation. Pick the hyperparameter pair with the lowest average 0-1 loss (classification error) and train an SVM with these hyperparameters using the complete training dataset. Only the training data must be used in the model selection process.

Report the values for C and γ you found in the model selection process, the 0-1 loss on the training data, as well as the 0-1 loss on the test data.

Deliverables: Description of software used; a short description of how you proceeded (e.g., did the cross-validation); training and test errors as well as the best hyperparameter configuration

2.3 Inspecting the kernel expansion

A support vector x_i is bounded if the corresponding coefficient in the kernel expansion (usually denoted by α_i) has an absolute value of C. If $0 < \alpha_i < C$ then the support vector is free. Free support vectors are those that lie on the margin (i.e., have a functional margin of 1). Bounded support vectors correspond to patterns having a functional margin smaller than 1. These include the wrongly classified patterns as well as the patterns that are correctly classified but are inside the margin area (i.e., they are too close to the decision boundary and have a functional margin less than 1).

Let us consider the effect of the regularization parameter C. How do you expect the number of bounded and free support vectors change if C is drastically increased and decreased? Briefly justify your claim.

Deliverables: Answer and argumentation

Optional, not for submission Count the number of free and bounded support vectors in your solution of the Parkinson exercise for different values of C. Does it support your claim above?

References

[1] C.-C. Chang and C.-J. Lin. LIBSVM: A library for support vector machines. *ACM Transactions Intelligent Systems Technology*, 2(3):27:1–27:27, 2011.

[2] A. Frank and A. Asuncion. UCI machine learning repository, 2010.

[3] C. Igel, T. Glasmachers, and V. Heidrich-Meisner. Shark. *Journal of Machine Learning Research*, 9:993–996, 2008.

[4] M. Little, P. McSharry, E. Hunter, J. Spielman, and L. Ramig. Suitability of dysphonia measurements for telemonitoring of Parkinson's disease. *IEEE Transactions on Biomedical Engineering*, 56(4):1015–1022, 2009.