Machine Learning 2020-2021

Home Assignment 4

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

The deadline for this assignment is **15 December 2020, 22:00**. 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 the PDF file.
- A .zip file with all your solution source 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. The programming language of the course is Python.
- IMPORTANT: Do NOT zip the PDF file, since zipped files cannot be opened in speed grader. Zipped pdf submissions will not be graded.
- Your PDF report should be self-sufficient. I.e., it should be possible to grade it without opening the .zip file. We do not guarantee opening the .zip file when grading.
- Your code should be structured such that there is one main file (or one main file per question) 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.
- Handwritten solutions will not be accepted, please use the provided latex template to write your report.

1 Airline Revisited (15 points)

We are back to the "airline" question from an earlier assignment. An airline has collected an i.i.d. sample of 10000 flight reservations and figured out that in this

sample 5 percent of passengers who made a reservation did not show up for the flight. They introduce a policy to sell 100 tickets for a flight that can hold only 99 passengers. Bound the probability of observing such sample and getting a flight overbooked.

We said that there are multiple ways to approach this question. This time we would like you to solve it in a specific way. (This time alternative solutions are not accepted.) Namely, consider the following process of generating the two samples:

1. We sample 10100 passenger show up events independently at random according to an unknown distribution p.
2. And then we split them into 10000 passengers in the collected sample and 100 passengers booked for the 99-seats flight.

Bound the probability of observing a sample of 10000 with 95% show ups and a 99-seats flight with all 100 passengers showing up by following the above sampling protocol. If you do things right, you can get a bound of about 0.0062 (there may be some variations depending on how exactly you do the calculation).

2 The Growth Function and the VC-Dimension (35 points)

1. Let $\mathcal{H}$ be a finite hypothesis set with $|\mathcal{H}| = M$ hypotheses. Prove that $m_{\mathcal{H}}(n) \leq \min\{M, 2^n\}$.
2. Bound the VC-dimension of $\mathcal{H}$ above (i.e., $|\mathcal{H}| = M$).
3. Let $\mathcal{H}$ be a hypothesis space with 2 hypotheses (i.e., $|\mathcal{H}| = 2$). Prove that $d_{\text{VC}}(\text{H}) = 1$.
4. Prove that $m_{\mathcal{H}}(2n) \leq m_{\mathcal{H}}(n)^2$.
5. What should be the relation between $d_{\text{VC}}(\text{H})$ and n in the VC generalization bound in Theorem 3.16 in Yevgeny's lecture notes in order for the bound to be non-trivial? [A bound on the loss that is greater than or equal to 1 is trivial, because we know that the loss is always bounded by 1. You do not have to make an exact calculation, giving an order of magnitude is sufficient.]
6. In the case of a finite hypothesis space, $|\mathcal{H}| = M$, compare the generalization bound that you can obtain with Theorem 3.16 in Yevgeny's lecture notes with the generalization bound in Theorem 3.2 in Yevgeny's lecture notes. What gives you a tighter bound?

7. How many samples do you need in order to estimate the loss of a linear classifier selected out of a set of linear classifiers in $\mathbb{R}^{10}$ within 0.01 precision with 99% confidence?

   Clarifications: (1) we want a "one-sided" estimation, i.e., we want that the expected loss of the classifier will not exceed the observed empirical loss by more than 0.01; (2) you are allowed to use the bound on the VC-dimension of linear classifiers mentioned in Yevgeny's lecture notes; (3) the selection is done based on the empirical loss of all the classifiers on the same sample; (4) solving the question analytically is a bit tricky, you are allowed to provide a numerical solution. In either case (numerical or analytical solution), please, explain clearly in your report what you did.

8. Let $\mathcal{H}_+$ be the class of "positive" circles in $\mathbb{R}^2$ (each $h \in \mathcal{H}_+$ is defined by the center of the circle $c \in \mathbb{R}^2$ and its radius $r \in \mathbb{R}$; all points inside the circle are labeled positively and outside negatively). Prove that $d_{VC}(\mathcal{H}_+) \geq 3$.

9. Let $\mathcal{H} = \mathcal{H}_+ \cup \mathcal{H}_-$ be the class of "positive" and "negative" circles in $\mathbb{R}^2$ (the "negative" circles are negative inside and positive outside). Prove that $d_{VC}(\mathcal{H}) \geq 4$.

10* ***Optional*** Prove the matching upper bounds $d_{VC}(\mathcal{H}_+) \leq 3$ and $d_{VC}(\mathcal{H}) \leq 4$. [Doing this formally is not easy, but will earn you extra honor.]

3 SVMs (50 points)

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.[1] You are supposed to use Python in this course, still we make some comments on SVM implementations available in other languages, which may be useful in other contexts. We might get back to this question in future assignments, so it may pay off to write clean, reusable code.

We recommend using LIBSVM Chang and Lin (2011), 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. The SVM implementation `sklearn.svm.SVC` (see `https://scikit-learn.org/stable/modules/svm.html`) of Scikit-learn Pedregosa et al. (2011) is also based

[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.

on LIBSVM.[2] 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 Little et al. (2009), which can be obtained from the well-known UCI benchmark repository Frank and Asuncion (2010).

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 Little et al. (2009), Frank and Asuncion (2010). 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 into a training and test set, `parkinsonsTrainStatML.dt` and `parkinsonsTestStatML.dt`, respectively.

3.1 Data normalization

Data normalization is an important preprocessing step. A basic normalization is to generate mean free, unit variance input data.

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.

[2] 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 Igel et al. (2008). Get the latest snapshot from `http://www.shark-ml.org/`.

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).

See also Example 5.3 on pages 174-175 in the course textbook Abu-Mostafa et al. (2012).

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

3.2 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*.

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*.

Cross validation is an important technique in machine learning and it is very important that you become familiar with it. You can read about cross-validation on pages 145-149 in the course textbook Abu-Mostafa et al. (2012).

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

3.3 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 lie on the margin (i.e., have a functional margin of 1). All 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) end up as bounded support vectors.

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 giving rigorous arguments.

In addition, verify your claim experimentally by count the number of free and bounded support vectors in your solution of the Parkinson exercise for different values of C. Report the C values and the (different) numbers of free and bounded support vectors. Show the part of the code that computes the number of free and bounded support vectors, respectively, in the report (in addition to the attached full source code in an archive).

Deliverables: Answer and rigorous argumentation, results of empirical validation including description of how these results were computed

References

Y. S. Abu-Mostafa, M. Magdon-Ismail, and H.-T. Lin. *Learning from Data*. AMLbook, 2012.

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.

A. Frank and A. Asuncion. UCI machine learning repository, 2010. URL `http://archive.ics.uci.edu/ml`.

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

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

F. Pedregosa, G. Varoquaux, A. Gramfort, V. Michel, B. Thirion, O. Grisel, M. Blondel, P. Prettenhofer, R. Weiss, V. Dubourg, J. Vanderplas, A. Passos, D. Cournapeau, M. Brucher, M. Perrot, and E. Duchesnay. Scikit-learn: Machine learning in Python. *Journal of Machine Learning Research*, 12:2825–2830, 2011.