Machine Learning B

2022-2023

Home Assignment 4

Yevgeny Seldin **Fabian Gieseke**
Department of Computer Science
University of Copenhagen

The deadline for this assignment is **20 December 2022, 18:00**. 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 full source code in the PDF file, only selected lines if you are asked to do so.
- 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.
- 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. 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 AdaBoost Example (25 points)

In the table provided below, you can find a simple data set consisting of data points $\mathbf{x} = [x_1, x_2]^{\mathrm{T}} \in \mathbb{R}^2$ and associated class labels $y_i \in \{-1, +1\}$. Your task is to apply the AdaBoost algorithm to this data set!

1. Fill the missing entries of the table (round the intermediate results up to three decimal numbers). Make use of:

   - $\varepsilon_i^{(b)} = w_i^{(b)} \mathbb{I}(h_b(\mathbf{x}_i) \neq y_i)$ and $\varepsilon_b = \sum_{j=1}^{n} \varepsilon_j^{(b)}$
   - $\gamma_i^{(b)} = \exp(-\alpha_b y_i h_b(\mathbf{x}_i))$ and $\Gamma^{(b)} = \sum_{j=1}^{n} w_j^{(b)} \gamma_j^{(b)}$

   Note that, for each of the boosting rounds, the weak learners h_i are already given (top row).

2. Provide the final model $h(\mathbf{x}) = \ldots$ and the induced predictions in the table (last column).

Data			$h_1(\mathbf{x}_i) = sign(x_1 - 0.25)$					$h_2(\mathbf{x}_i) = sign(-x_2 + 0.55)$					$h_3(\mathbf{x}_i) = sign(x_2 - 0.35)$					Final
x_1	x_2	y	$w_i^{(1)}$	$h_1(\mathbf{x}_i)$	$\varepsilon_i^{(1)}$	$\gamma_i^{(1)}$	$w_i^{(1)}\gamma_i^{(1)}$	$w_i^{(2)}$	$h_2(\mathbf{x}_i)$	$\varepsilon_i^{(2)}$	$\gamma_i^{(2)}$	$w_i^{(2)}\gamma_i^{(2)}$	$w_i^{(3)}$	$h_3(\mathbf{x}_i)$	$\varepsilon_i^{(3)}$	$\gamma_i^{(3)}$	$w_i^{(3)}\gamma_i^{(3)}$	$h(\mathbf{x}_i)$
0.1	0.15	-1	0.1															
0.15	0.25	-1	0.1															
0.3	0.7	-1	0.1															
0.6	0.6	-1	0.1															
0.9	0.3	-1	0.1															
0.3	0.1	1	0.1															
0.35	0.45	1	0.1															
0.45	0.1	1	0.1															
0.45	0.5	1	0.1															
0.6	0.4	1	0.1															
			$\varepsilon_1 =$		$\alpha_1 =$		$\Gamma^{(1)} =$	$\varepsilon_2 =$		$\alpha_2 =$		$\Gamma^{(2)} =$	$\varepsilon_3 =$		$\alpha_3 =$		$\Gamma^{(3)} =$	

2 Transforming AdaBoost Weights (25 points)

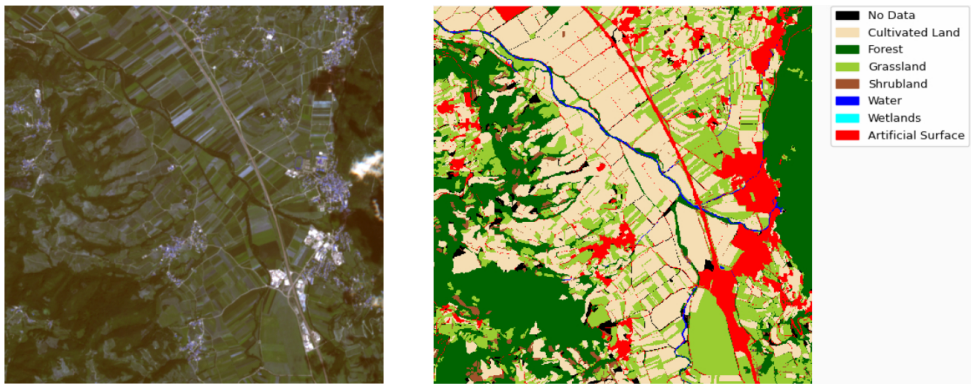
Assume you are given a training set $T = \{(\mathbf{x}_1, y_1), \ldots, (\mathbf{x}_n, y_n)\} \subset \mathbb{R}^d \times \{-1, +1\}$ of labelled instances. Prove, by induction over b, that the weight updates

$$w_i^{(b+1)} = \frac{w_i^{(b)} \exp(-\alpha_b y_i h_b(\mathbf{x}_i))}{\sum_{j=1}^{n} w_j^{(b)} \exp(-\alpha_b y_j h_b(\mathbf{x}_j))}$$

introduced in Adaboost can be written as

$$w_i^{(b+1)} = \frac{\exp(-y_i f_b(\mathbf{x}_i))}{\sum_{j=1}^{n} \exp(-y_j f_b(\mathbf{x}_j))},$$

where h_b is the weak learner fitted in boosting round b, α_b the importance of h_b, and $f_b = \sum_{p \leq b} \alpha_p h_p$.



(a) Image Data (b) Landcover Classification

Figure 1: The left image shows some input data. Here, only an RGB image is shown, but typically, more than three "bands" are available (e.g., RGB + near-infrared). The right image shows the classification of each pixel into one of the classes ("no data" corresponds to missing data). Such a plot is called landcover map.

3 Landcover Classification (35 points)

An important problem in remote sensing is landcover classification, i.e., the pixel-wise classification of satellite image data, see Figure 1. Often, multiple input images are given for a location, which are acquired over time. For instance, the popular Sentinel satellites[1] have a revisit time of 5 days, i.e., they take one image every 5 days for each location that is monitored. Manually labelling billions of such pixels is too time-consuming and machine learning models are often used in this context to (automatically) obtain such landcover classifications.

In this exercise, you are supposed to apply AdaBoost to such a data set. As input data, you are given two files, `train.npz` and `test.npz`, containing some training and some test instances.[2] Each instance is composed of images of size 13×13 pixels from 6 "bands" (i.e., channels) and 12 timestamps, along with a class label corresponding to the (landcover) class corresponding to the central pixel. A simple way to obtain a decent landcover classification model is to consider these image data as simple feature vectors and to make use of tree ensemble models such as boosted trees (in fact, this has been the standard for many years before the deep learning wave). The Jupyter notebook `LandcoverClassification.ipynb` already contains some code to load the two datasets and to visualize the data. Extend this notebook and conduct the following steps:

[1] `https://sentinel.esa.int/web/sentinel/missions`

[2] You can download both files as well as a Jupyter notebook from: `https://sid.erda.dk/cgi-sid/ls.py?share_id=c9SgMJSGik`

1. Considering all the pixels as input features can quickly become time-consuming for AdaBoost. Select only the central pixels per image as features (the resulting feature vectors should consist of 72 values). Argue why this is a decent choice to accelerate the fitting process for the task at hand. Also argue why one might "loose" something by doing so.

2. Conduct 2-fold cross validation on the training data to select the best-performing AdaBoost model. As weak learners/models, consider simple decision tree classifiers with the Gini index as impurity measure. For AdaBoost, make use of the 'SAMME' discrete boosting approach that you know from the lecture. As grid, consider:

   - *Depth of decision trees:* [1,2,3]
   - *Number of boosted trees:* [50,100,200]

   After having performed the grid search, select the best model, refit the model to the entire training data, and compute the test accuracy. Also generate a confusion matrix on the test data to visualize the errors made by the model. What is the accuracy of the model on the test set?

3. Optional: Can you improve the quality of the model by considering other features and/or other assignments for the parameters?

Hint: Make use of Scikit-Learn and the `GridSearchCV` class. You can provide a base estimator to the AdaBoost class. Here, figure out how to provide the tree depth as parameter to the grid search.

4 PAC-Bayes vs. Occam (15 points)

The change of measure inequality that is at the basis of PAC-Bayesian analysis can be seen as a replacement of the union bound, which is at the basis of Occam's razor. In this question we compare the tightness of the two approaches.

1. Prove the following theorem.

   Theorem 1. Let S be an i.i.d. sample of n points, let ℓ be the zero-one loss, let $\mathcal{H}$ be countable, and let $\pi(h)$ be such that it is independent of the sample S and satisfies $\sum_{h\in\mathcal{H}} \pi(h) \leq 1$. Let $\delta \in (0,1)$. Then with probability greater than $1-\delta$, for all distributions ρ over $\mathcal{H}$ simultaneously:

$$\mathrm{kl}\left(\mathbb{E}_\rho\left[\hat{L}(h,S)\right] \middle\| \mathbb{E}_\rho\left[L(h)\right]\right) \leq \frac{\mathbb{E}_\rho\left[\ln \frac{1}{\pi(h)}\right] + \ln \frac{n+1}{\delta}}{n}. \tag{1}$$

You can use Occam's razor bound based on kl inequality that you have derived in one of the earlier questions to prove Theorem 1.

2. Recall that by PAC-Bayes-kl inequality, under the conditions of Theorem 1 we have that with probability greater than $1-\delta$, for all distributions ρ over $\mathcal{H}$ simultaneously:

$$\mathrm{kl}\left(\mathbb{E}_\rho\left[\hat{L}(h,S)\right]\middle\|\mathbb{E}_\rho\left[L(h)\right]\right) \leq \frac{\mathrm{KL}(\rho\|\pi)+\ln\frac{n+1}{\delta}}{n}. \tag{2}$$

Show that the PAC-Bayes-kl inequality in equation (2) is always at least as tight as the Occam's razor bound with kl in equation (1). Hint: the entropy of a discrete distribution is always non-negative, $\mathrm{H}(\rho) \geq 0$. Recall the definition of $\mathrm{KL}(\rho\|\pi)$.

5 [Optional question] Regularization by the relative entropy and the Gibbs distribution

In this question we will show that regularization by relative entropy leads to solutions in a form of the Gibbs distribution. Let's assume that we have a finite hypothesis class $\mathcal{H}$ of size m and we want to minimize

$$\mathcal{F}(\rho) = \alpha\mathbb{E}_\rho\left[\hat{L}(h,S)\right] + \mathrm{KL}(\rho\|\pi) = \alpha\sum_{h=1}^{m}\rho(h)\hat{L}(h,S) + \sum_{h=1}^{m}\rho(h)\ln\frac{\rho(h)}{\pi(h)}$$

with respect to the distribution ρ. This objective is closely related to the objective of PAC-Bayes-λ inequality when λ is fixed and this sort of minimization problem appears in many other places in machine learning. Let's slightly simplify and formalize the problem. Let $\rho = (\rho_1, \ldots, \rho_m)$ be the posterior distribution, $\pi = (\pi_1, \ldots, \pi_m)$ the prior distribution, and $L = (L_1, \ldots, L_m)$ the vector of losses. You should solve

$$\begin{aligned} \min_{\rho_1,\ldots,\rho_m} \quad & \alpha\sum_{h=1}^{m}\rho_h L_h + \sum_{h=1}^{m}\rho_h\ln\frac{\rho_h}{\pi_h} \\ s.t. \quad & \sum_{h=1}^{m}\rho_h = 1 \\ & \forall h : \rho_h \geq 0 \end{aligned} \tag{3}$$

and show that the solution is $\rho_h = \frac{\pi_h e^{-\alpha L_h}}{\sum_{h'=1}^{m}\pi_{h'}e^{-\alpha L_{h'}}}$. Distribution of this form is known as the Gibbs distribution.

Guidelines:

1. Instead of solving minimization problem (3), solve the following minimization problem

$$\begin{aligned} \min_{\rho_1,\ldots,\rho_m} \quad & \alpha \sum_{h=1}^{m} \rho_h L_h + \sum_{h=1}^{m} \rho_h \ln \frac{\rho_h}{\pi_h} \\ s.t. \quad & \sum_{h=1}^{m} \rho_h = 1, \end{aligned} \tag{4}$$

   i.e., drop the last constraint in (3).

2. Use the method of Lagrange multipliers to show that the solution of the above problem has a form of $\rho_h = \pi_h e^{-\alpha L_h + \texttt{something}}$, where `something` is something involving the Lagrange multiplier.

3. Show that $\rho_h \geq 0$ for all h. (This is trivial. But it gives us that the solutions of (3) and (4) are identical.)

4. Finally, $e^{\texttt{something}}$ should be such that the constraint $\sum_{h=1}^{m} \rho_h = 1$ is satisfied. So you can easily get the solution. You even do not have to compute the Lagrange multiplier explicitly.