Machine Learning A

2024-2025

Home Assignment 3

Christian Igel **Sadegh Talebi**
Department of Computer Science
University of Copenhagen

The deadline for this assignment is **19 September 2024, 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 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.
- IMPORTANT: Do NOT zip the PDF file, since zipped files cannot be opened in *SpeedGrader*. 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 Preprocessing (33 points)

To solve this exercise, it is helpful to read Section 9.1 in e-Chapter 9 of (Abu-Mostafa et al., 2015). It is also recommended to read Section 4.2 of the textbook (Abu-Mostafa et al., 2015) on regularization. The e-Chapter 9 can be downloaded from `https://amlbook.com/eChapters.html` or via Absalon. Note that the *in-sample error* E_{in} corresponds to what we call the empirical risk (or training error).

1.1 Importance of Preprocessing (6 points)

Each client at the Bank of Learning (BoL) is described by the two-dimensional feature vector (Age, Income). The bank gave two persons A and B credit cards based on their age and income.

Person	Age in years	Income in thousands of USD
A	47	35
B	22	40

A paid off her credit card bill, but B defaulted. Now consider C who has 'coordinates' $(21, 36)$ applies for credit. Should the BoL give him credit, according to the nearest neighbor algorithm? If income is measured in 'dollars' instead of in 'thousands of dollars', what is your answer?

1.2 Input Centering (9 points)

Let $\boldsymbol{X} \in \mathbb{R}^{N \times d}$ be the data matrix, whose rows are formed the input data vectors $\boldsymbol{x}_1, \dots, \boldsymbol{x}_N$, all living in $\mathbb{R}^d$ (not augmented with a 1):

$$\boldsymbol{X} = \begin{bmatrix} \text{——} \boldsymbol{x}_1^\top \text{——} \\ \text{——} \boldsymbol{x}_2^\top \text{——} \\ \vdots \\ \text{——} \boldsymbol{x}_N^\top \text{——} \end{bmatrix}$$

Let $\bar{\boldsymbol{x}}$ be the *in-sample mean vector* of the input data, defined as $\bar{\boldsymbol{x}} = \frac{1}{N}\sum_{n=1}^{N} \boldsymbol{x}_n$; in matrix notation, $\bar{\boldsymbol{x}} = \frac{1}{N}\boldsymbol{X}^\top \mathbf{1}$, where $\mathbf{1}$ is a vector of all ones. To obtain the transformed vector, we simply subtract the mean from each data point: $\boldsymbol{z}_n = \boldsymbol{x}_n - \bar{\boldsymbol{x}}$ for $n = 1, \dots, N$. Let $\boldsymbol{Z}$ be the matrix collecting $\boldsymbol{z}_1, \dots, \boldsymbol{z}_N$, constructed similarly to $\boldsymbol{X}$.

(a) Define $\gamma = \boldsymbol{I} - \frac{1}{N}\boldsymbol{x}\boldsymbol{x}^\top$, where $\boldsymbol{I}$ denotes the identity matrix. Show that $\boldsymbol{Z} = \gamma \boldsymbol{X}$.

It is worth noting that γ is called the centering operator, which projects onto the space orthogonal to $\mathbf{1}$. You may read further in Appendix B.3 of Abu-Mostafa et al. (2015).

(b) Show that the rank of $\boldsymbol{Z}$ is smaller than d. You may presume the fundamental result that the column rank and the row rank are always equal.

1.3 Input Whitening (18 points)

Let $\hat{x}_1$ and $\hat{x}_2$ be independent with zero mean and unit variance. You measure inputs $x_1 = \hat{x}_1$ and $x_2 = \sqrt{1-\epsilon^2}\hat{x}_1 + \epsilon x_2$. Let $\boldsymbol{x} = (x_1, x_2)^\mathrm{T}$ and $\hat{\boldsymbol{x}} = (\hat{x}_1, \hat{x}_2)^\mathrm{T}$.

(a) What are $\mathrm{Var}(x_1)$, $\mathrm{Var}(x_1)$, and $\mathrm{Cov}(x_1, x_2)$? (Here, 'Var' and 'Cov' denotes the variance and covariance, respectively.)

(b) Suppose $f(\hat{\boldsymbol{x}}) = \hat{w}_1\hat{x}_1 + \hat{w}_2\hat{x}_2$ (linear in the independent variables). Show that f is linear in the correlated inputs, $f(\boldsymbol{x}) = w_1x_1 + w_2x_2$. (Obtain w_1, w_2 as functions of $\hat{w}_1$, $\hat{w}_2$.)

(c) Consider the 'simple' target function $f(\hat{\boldsymbol{x}}) = \hat{x}_1 + \hat{x}_2$. If you perform regression with the correlated inputs $\boldsymbol{x}$ (i.e., your model is of the form $f(\boldsymbol{x}) = w_1x_1 + w_2x_2$) and regularization constraint $w_1^2 + w_2^2 \le C$, what is the maximum amount of regularization you can use (i.e., the minimum value of C) and still be able to implement the target?

(d) What happens to the minimum C as the correlation increases (i.e., $\epsilon \to 0$)?

Note: The question is adopted from Exercise 9.4 on page e-Chap:9–4 in the textbook by Abu-Mostafa et al. (2015). As in the book, the notation is overloaded. Part (b) could be rephrased as: Let $\boldsymbol{x} = (x_1, x_2)^\mathrm{T}$ depend on $\hat{\boldsymbol{x}} = (\hat{x}_1, \hat{x}_2)^\mathrm{T}$ as defined above. Let $f(\hat{\boldsymbol{x}}) = \hat{w}_1\hat{x}_1 + \hat{w}_2\hat{x}_2$ for some $w_1, w_2 \in \mathbb{R}$. Show that there exists a function $g(\boldsymbol{x}) = w_1x_1 + w_2x_2$ such that $f(\hat{\boldsymbol{x}}) = g(\boldsymbol{x})$.

Hints: Recall that for random variables X, Y, V, W and scalars a, b, c, d:

$$\mathrm{Var}(aX + bY) = a^2\,\mathrm{Var}(X) + b^2\,\mathrm{Var}(Y) + a^2\,\mathrm{Cov}(X, Y) \tag{1}$$

$$\begin{aligned}\mathrm{Cov}(aX + bY, cV + dW) = {} & ac\,\mathrm{Cov}(X, V) + ad\,\mathrm{Cov}(X, W) \\ & + bc\,\mathrm{Cov}(Y, V) + bd\,\mathrm{Cov}(Y, W)\end{aligned} \tag{2}$$

2 Competition Design to Find Defective Products (24 points)

1. Suppose you are working at a factory and you are assigned a task to identify defective products having a certain fault. To this effect, you have collected an i.i.d. sample of 3130 products from the inventory and annotated it for presence or absence of the fault —hence, a binary annotation.

   You organize a competition to find a classifier for the defective product. Suppose that 20 teams have signed up for the competition and your boss requires you to provide a performance guarantee on the prediction accuracy of the best classifier that does not deviate by more than 0.04 from its true performance with probability exceeding 98%. In other words, with probability greater than 98%, the estimate of the expected error should not underestimate the true expected zero-one error of the selected classifier by more than 0.04 (one-sided error).

   How many samples do you have to keep aside in order to satisfy this requirement, assuming that you accept 1 solution from each team? Provide a complete calculation.

   Beware that numerical answers without any derivations or explanations will not be accepted.

2. Assume you have conducted the competition above, but were not satisfied with the prediction accuracy of the winning team's classifier. You decided to make another competition and were very lucky to convince your boss to support annotation of another 1800 products. You decided to release the old data containing 3130 products for training and keep the new 1800 samples for evaluating the outcome of the new competition. Your boss requires from you the same performance guarantee as before.

   How many teams can you accept to take part in the competition assuming that you accept only 1 solution from each team? Provide a complete calculation.

   Beware that numerical answers without any derivations or explanations will not be accepted.

3 Combining Multiple Confidence Intervals (22 points)

Alex works at a pharmaceutical company. She has developed a new treatment and is interested in estimating its success probability μ. To this end, she has collected an i.i.d. sample S, from which she computed the sample mean $\widehat{\mu}$, as an estimate of μ. However, the company requires a confidence interval (CI) for μ too.

Rather than computing the CI herself, she decided to outsource the task to three data science companies, who are world-class experts in this area and have developed advanced tools to compute confidence intervals. The interaction protocol with each company $i \in \{1, 2, 3\}$ is as follows:

- The company i receives a dataset S_i as well as a parameter δ_i as input, and returns $\texttt{CI}_i = [l_i, u_i]$ that is a $(1 - \delta_i)$-CI for the mean of the distribution underlying S_i.
- The company does not reveal how they build $\texttt{CI}_i$, and hence the 3 companies may use different methods.

Assume that the project requires a 0.99-CI on μ and that Alex decides to release the entire dataset S to each company —i.e., $S_i = S, i = 1, \dots, 3$.

Explain how Alex must choose $\delta_i, i = 1, \dots, 3$ when contracting each company and how she must use the CIs received from the companies (i.e., what is the best CI she can report to her boss).

4 Early Stopping (21 points)

Early stopping is a widely used technique to avoid overfitting in models trained by iterative methods, such as gradient descent. In particular, it is used to avoid overfitting in training neural networks. In this question we analyze several ways of implementing early stopping. The technique sets aside a validation set S_{val}, which is used to monitor the improvement of the training process. Let $h_1, h_2, h_3, \dots$ be a sequence of models obtained after $1, 2, 3, \dots$ epochs of training a neural network or any other prediction model. (You do not need to know any details about neural networks or their training procedure to answer the question.) Let $\hat{L}(h_1, S_{\text{val}}),\ \hat{L}(h_2, S_{\text{val}}),\ \hat{L}(h_3, S_{\text{val}}), \dots$ be the corresponding sequence of validation errors on the validation set S_{val}.

1. Let h_{t^*} be the neural network returned after training with early stopping. In which of the following cases $\hat{L}(h_{t^*}, S_{\text{val}})$ is an unbiased estimate of $L(h_{t^*})$ and in which cases it is not? Please, explain your answer.

   (a) *Predefined Stopping:* the training procedure always stops after 100 epochs and always returns the last model $h_{t^*} = h_{100}$.

(b) *Non-adaptive Stopping:* the training procedure is executed for a fixed number of epochs T, and returns the model h_{t^*} with the lowest validation error observed during the training process, i.e., $t^* = \arg\min_{t \in \{1,\dots,T\}} \hat{L}(h_t, S_{\text{val}})$.

(c) *Adaptive Stopping:* the training procedure stops when no improvement in $\hat{L}(h_t, S_{\text{val}})$ is observed for a significant number of epochs. It then returns the best model observed ever during training. (This procedure is proposed in Goodfellow et al. (2016, Algorithm 7.1) or `https://www.quora.com/How-does-one-employ-early-stopping-in-TensorFlow`, but again, you do not need to know the details of the training procedure.)

2. Derive a high-probability bound (a bound that holds with probability at least $1 - \delta$) on $L(h_{t^*})$ in terms of $\hat{L}(h_{t^*}, S_{\text{val}})$, δ, and the size n of the validation set S_{val} for the cases Predefined Stopping (Part 1.a) and Non-adaptive Stopping (1.b). In the second case, the bound may additionally depend on the total number of epochs T.

Note: Additional questions from past home assignments or exams, for further practice.

5 How to split a sample into training and tests set (Optional, 0 points)

In this question you will analyze one possible approach to the question of how to split a dataset S into training and test sets, $S^{\texttt{train}}$ and $S^{\texttt{test}}$. As we have already discussed, overly small test sets lead to unreliable loss estimates, whereas overly large test sets leave too little data for training, thus producing poor prediction models. The optimal trade-off depends on the data and the prediction model. So can we let the data speak for itself? We will give it a try.

We want to find a good balance between the sizes of $S^{\texttt{train}}$ and $S^{\texttt{test}}$. We consider m possible splits $\{(S_1^{\texttt{train}}, S_1^{\texttt{test}}), \dots, (S_m^{\texttt{train}}, S_m^{\texttt{test}})\}$, where the sizes of the test sets are $n_1, \dots, n_m$, correspondingly. For example, it could be $(10\%, 90\%), (20\%, 80\%), \dots, (90\%, 10\%)$ splits or anything else with a reasonable coverage of the possible options. We train m prediction models $\hat{h}_1^*, \dots, \hat{h}_m^*$, where $\hat{h}_i^*$ is trained on $S_i^{\texttt{train}}$. We calculate the test loss of the i-th model on the i-th test set $\hat{L}(\hat{h}_i^*, S_i^{\texttt{test}})$. Derive a bound on $L(\hat{h}_i^*)$ in terms of $\hat{L}(\hat{h}_i^*, S_i^{\texttt{test}})$ and n_i that holds for all $\hat{h}_i^*$ simultaneously with probability at least $1 - \delta$.

Comment: No theorem from the lecture notes applies directly to this setting, because they all have a fixed sample size n, whereas here the sample sizes $n_1, \dots, n_m$ vary. You have to provide a complete derivation.

6 Efficient use of data (Optional, 0 points)

So far, given a data set S, in all our theoretical results we used part of the data for training prediction rules and another part for validating them. This way some data are only used for training and some data are only used for validation. But put attention that if we would have trained a prediction rule on the validation set and validated it on the training set, we would have also gotten an unbiased estimate of the loss. (Remember: "it's not about how you call it, it's about how you use it"!). So could we use the data more efficiently?

The approach of using part of the data for training and part for validation, and then reverting the roles, somewhat resembles cross-validation. But a big word of warning would be in place here. Even though the standard cross-validation technique is widely used, it is a heuristic, and if it is used to validate too many prediction rules, it is prone to overfitting, in exactly the same way as the standard validation technique is prone to overfitting, unless generalization bounds are used to control the overfitting.

What you will do next is inspired by cross-validation, but it is different from the standard cross-validation approach.

So, we have a data set S of size n (assume that n is even). We split the data set into two equal halves, $S = S_0 \cup S_1$. We train M models $\{h_{0,1}, \dots, h_{0,M}\}$ on the first half of the data and validate them on the remaining half. Let $\hat{L}(h_{0,i}, S_1)$ for $i \in \{1, \dots, M\}$ be the corresponding validation losses. Then we train another M models $\{h_{1,1}, \dots, h_{1,M}\}$ on the second half of the data and validate them on the first half. Let $\hat{L}(h_{1,i}, S_0)$ for $i \in \{1, \dots, M\}$ be the corresponding validation losses. Finally, we select the model $h_{j^*,i^*} = \arg \min_{j \in \{0,1\}, i \in \{1,\dots,M\}} \hat{L}(h_{j,i}, S_{1-j})$ with the smallest validation loss.

Derive a high-probability generalization bound for the expected loss of h_{j^*,i^*}. (I.e., a bound on $L(h_{j^*,i^*})$ that holds with probability at least $1 - \delta$.)

Comment: no theorem in the lecture notes directly applies to the question, because they all assume that $\hat{L}(h, S)$ is computed on the same S for all h. You have to make a custom derivation, but it will not be very different from derivations you can find in the lecture notes.

References

Yaser S. Abu-Mostafa, Malik Magdon-Ismail, and Hsuan-Tien Lin. *Learning from Data. Dynamic E-Chapters*. AMLbook, 2015.

Ian Goodfellow, Yoshua Bengio, and Aaron Courville. *Deep Learning*. MIT Press, 2016. `http://www.deeplearningbook.org`.