Machine Learning A

2022-2023

Final Exam

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

You must submit your individual solution of the exam electronically via the **Digital Exam / Digital Eksamen** system. The deadline for submitting the exam is **Friday, 4 November 2022, at 16:00**. The exam must be solved **individually**. You are **not allowed** to work in groups or discuss the exam questions with other students. For fairness reasons any questions about the exam will be answered on Absalon. If your question may reveal the answer to other students, please, email it personally to the lecturers and we will either answer it on Absalon or tell you that we cannot answer your question.

WARNING: The goal of the exam is to evaluate your personal achievements in the course. We believe that take-home exams are most suitable for this evaluation, because they allow to test both theoretical and practical skills. However, our ability to give take-home exams strongly depends on your honesty. Therefore, any suspicion of cheating, in particular collaboration with other students, will be directly reported to the head of studies and prosecuted in the strictest possible way. It is also strictly prohibited to post the exam questions or parts thereof on the Internet or on discussion forums and to seek help on discussion forums. And you are not allowed to store your solutions in open access version control repositories or to post them on the Internet or on discussion forums. Be aware that if proven guilty you may be expelled from the university. Do not put yourself and your fellow students at risk.

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. Do *not* include the task description or parts thereof in your report.
- Please, use the provided LaTeX template for typing your report. Hand-

written solutions will not be accepted.

- 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.
- 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 run your program.

1 The Basic Assumptions (5 points) [Yevgeny]

1. What are the two basic assumptions that are behind all supervised learning algorithms taught in this course?
2. For each of the two assumptions explain why it is necessary and what might potentially go wrong if it is violated. (You are allowed to consider extreme violations for the purpose of illustration.)

2 Weighted Neighbors (15 points) [Yevgeny]

We have studied the K-nearest neighbor algorithm, which predicts a label Y of a point X by taking the majority label of the K nearest neighbors of X in a sample. An alternative is to take a weighted majority label, where the weight would decrease with the distance of the sample points from the target point. In other words, assuming that $\mathcal{Y} = \{\pm 1\}$, weighted neighbors predict

$$h_{f,d}(X) = \operatorname{sign}\left(\sum_{i=1}^{n} \frac{f(d(X, X_i))}{\sum_{i'=1}^{n} f(d(X, X_{i'}))} Y_i\right),$$

where $d(X, X_i)$ is the distance between the target point X and a sample point X_i, and $f(d(X, X_i))$ is a decreasing function of the distance, with $\frac{f(d(X,X_i))}{\sum_{i'=1}^{n} f(d(X,X_{i'}))}$ being the weight of the sample point (X_i, Y_i) in the weighted decision on the label of the target point X. (It is easy to see that the weights sum up to 1.)

The question of how to select K in K-nearest neighbors is now replaced by the question of how fast should be the decrease of f. If f is decreasing fast, only few nearest neighbors will have a non-negligible weight, and the prediction rule will be similar to K-nearest neighbors with small K. If f is decreasing slowly, then neighbors that are further away will also have an impact, making the prediction rule similar to K-nearest neighbors with a large K. In the extreme case when f is constant (not decreasing at all), all points will have the same weight and the prediction rule with be identical to K-NN with $K = n$.

Your goal in this question is to derive generalization bounds for selection of f and d. We assume that we use a sample S_1 of size n_1 to generate a weighted neighbors prediction rule (the sample from which we take the neighbors) and a sample S_2 of size n_2 to compute its empirical error.

1. Assume that the distance measure d is fixed. Let $f_i(z) = \exp(-2^i z)$ for $i \in \{\dots, -2, -1, 0, 1, 2, \dots\}$ (all the integer numbers). Derive a generalization bound for $L(h_{f_i,d})$ that holds for all i simultaneously.
2. Now assume that we have an infinite set of distance measures $d_1, d_2, \dots$, and let h_{f_i,d_j} be a weighted neighbors prediction rule with distance measure d_j and weight decay f_i. Derive a generalization bound for $L(h_{f_i,d_j})$ that holds for all i and j simultaneously.

3 The Impact of Dependence (20 points) [Yevgeny]

We have said many times during the course that independence of training samples is crucial for learning. In one of the home assignments you have also constructed an example of dependent random variables for which the empirical mean does not converge to the expected value as the number of variables grows. In this question we study the impact of partial dependence on generalization.

Imagine that you have collected an annotated sample of n points, $S = \{(X_1, Y_1), \dots, (X_n, Y_n)\}$, and you know that there is dependence, which has the following structure: the sample is split into subsets of r points each, where the points within the subsets are dependent, but the points across the subsets are independent. For example, you have selected $n/4$ households at random, each household including 4 members ($r = 4$), and then performed some test on all members of each household, say, a covid test. Assume that n is a multiple of r. The results for all members of a given household are dependent, but the results across households are independent. We assume that each r consecutive samples are the dependent ones, i.e., $S = \{\{(X_1, Y_1), \dots, (X_r, Y_r)\}, \{(X_{r+1}, Y_{r+1}), \dots, (X_{2r}, Y_{2r})\}, \dots\}$, where the subsets in the curly brackets are dependent. We assume that all (X_i, Y_i) pairs come from the same distribution.

1. One possible approach to cope with dependence is to take a subset of samples $S' = \{(X_1, Y_1), (X_{r+1}, Y_{r+1}), (X_{2r+1}, Y_{2r+1}), \dots\}$, where we take a single sample from each dependent subset. Then the samples in S' are independent. Let $\hat{L}(h, S') = \frac{r}{n} \sum_{i=0}^{(n/r)-1} \ell(h(X_{ir+1}), Y_{ir+1})$ be the empirical loss of a prediction rule h on S'. Let $\mathcal{H}$ be a hypothesis class with $|\mathcal{H}| = M$. Derive a generalization bound for $L(h)$ in terms of $\hat{L}(h, S')$, that holds for all $h \in \mathcal{H}$.

2. The above approach wastes a lot of samples. An alternative approach is to use all the samples. Let $\hat{L}(h, S) = \frac{1}{n} \sum_{i=1}^{n} \ell(h(X_i), Y_i)$ be the empirical loss of h on all of S. Let $\mathcal{H}$ be as before. Derive a generalization bound for $L(h)$ in terms of $\hat{L}(h, S)$, that holds for all $h \in \mathcal{H}$.

3. Did you get an improvement by using $\hat{L}(h, S)$ instead of $\hat{L}(h, S')$ or not? Explain why you got an improvement or why you did not get it.

4 Sleep Well (30 points) [Christian]

Sleep is one of the most fundamental physiological processes, and abnormal sleeping patterns are associated with poor health. They may, for example, indicate brain- & heart diseases, obesity and diabetes. During sleep our brain goes through a series of changes between different *sleep stages*, which are characterized by specific brain and body activity patterns. *Sleep staging* refers to the process of mapping these transitions over a night of sleep. This is of fundamental importance in sleep medicine, because the sleep patterns combined with other variables provide the basis for diagnosing many sleep related disorders (Kales and Rechtschaffen, 1968, Iber and AASM, 2007). The stages can be determined by measuring the neuronal activity in the cerebral cortex (via electroencephalography, EEG), eye movements (via electrooculography, EOG), and/or the activity of facial muscles (via electromyography, EMG) in a *polysomnography* (PSG) study. The classification into stages is done manually. This is a difficult and time-consuming process, in which expert clinicians inspect and segment the typically 8–24 hours long multi-channel signals. Contiguous, fixed-length intervals of 30 seconds are considered, and each of these *segments* is classified individually. Algorithmic sleep staging aims at automating this process. The state-of-the-art in algorithmic sleep staging is marked by deep neural networks, which can be highly accurate and robust, even compared to human performance, see the recent work by Perslev et al. (2019) and references therein.

This assignment considers algorithmic sleep staging. The data is based on a single EEG channel from the Sleep-EDF-15 data set (Kemp et al., 2000, Goldberger et al., 2000). The input is given by an intermediate representation from the U-Time neural network architecture (Perslev et al., 2019), the targets are sleep stages. We created a training and test set, the inputs and the corresponding labels can be found in `X_train.csv` and `y_train.csv` and `X_test.csv` and `y_test.csv`, respectively.

4.1 Data understanding and preprocessing

Download and extract the data.

Consider the training data `X_train.csv` and the corresponding labels `y_train.csv`. Report the class frequencies, that is, for each of the 5 classes report the number of data points divided by the total number of data points) in the training data.

The ith row in `X_train.csv` are the features of the ith training pattern. The class label of the ith pattern is given in the ith row of `y_train.csv`.

Deliverables: description of software used; frequency of classes

4.2 Principal component analysis

Perform a principal component analysis of the training data `X_train.csv`. Plot the eigenspectrum (see the plot on slide 28 of the *PCA* slides for an example). How many components are necessary to "explain 90 % of the variance"? Visualize the data by a scatter plot of the data projected on the first two principal components. Use different colors for the different classes in the plot.

Deliverables: description of software used; plot of the eigenspectrum; number of components necessary to explain 90 % of variance; scatter plot of the data projected on the first two principal components with different colors indicating the 5 different classes

4.3 Clustering

Perform 5-means clustering of `X_train.csv`. After that, project the cluster centers to the first two principal components of the training data. Then visualize the clusters by adding the cluster centers to the plot from the previous exercise.

Briefly discuss the results.[1]

Deliverables: description of software used; one plot with cluster centers and data points; short discussion of results

[1]In this example, you do not get 5 nicely separated clusters in 2D. A better looking projection of the data can be achieved using a non-linear dimensionality reduction technique, for example *non-linear t-Distributed Stochastic Neighbor Embedding* (t-SNE) (van der Maaten and Hinton, 2008). Although not part of the exam, we recommend that you try it out.

4.4 Classification

The task is to evaluate several multi-class classifiers on the data. Build the models using the training data only. The test data must only be used for final evaluation.

1. Apply multi-nominal logistic regression. If you use regularization, describe the type of regularization you used. Report training and test loss (in terms of 0-1 loss).
2. Apply random forests with 50, 100, and 200 trees. Report training and test loss (in terms of 0-1 loss).
3. Apply k-nearest-neighbor classification. Use cross-validation to determine the number of neighbors. Report training and test loss (in terms of 0-1 loss). Describe how you determined the number of neighbors.

Deliverables: description of software used; training and test errors; description of regularization and model selection process

5 Thanks for the Fish (15 points) [Christian]

Background

Consider the data in the files `smallFishInput.dt` and `smallFishLabel.dt`. They represent the length of a species of fish as a function of age and water temperature. The fish were kept in tanks at 25, 27, 29 and 31 degrees Celsius. After birth, a test specimen was chosen at random every 14 days and its length was measured. The dependent (output) variable in `smallFishLabel.dt`. is the length of the fish. The inputs are the age of the fish in days and the water temperature in degrees Celsius, stored in the corresponding lines in `smallFishInput.dt` . The data are described in the textbooks by Freund and Minton (1979) and Späth (1991). Of course, the length of a fish is a non-linear function of the animal's age. It saturates when the fish is full-grown. In this exercise, we consider the period of growth and investigate if the growth in length can be modelled by a linear function. Therefore, animals older than 90 days were discarded from the original data set.

Task: Linear regression

The goal of modeling is to find a mapping $f : \mathbb{R}^2 \to \mathbb{R}$. The predictor (input) variables are in `smallFishInput.dt` and the response (output) variable in `smallFishLabel.dt`.

Build an *affine* linear model of the data using linear regression. Report the three parameters of the model as well as the mean-squared-error of the model computed over the complete data set.

Compare the mean-squared-error with the variance of the data. Discuss based on this comparison whether you got a good fit or not.

Deliverables: description of software used; mean-squared error; parameters of the regression model; brief discussion relating mean-squared-error to the biased sample variance

6 Clustering (15 points) [Sadegh]

Consider a dataset consisting of 6 data points A, B, C, D, E, F as shown in Figure 1. The pair next to each point shows the x and y coordinates of each data point. We wish to group the datapoints into k clusters according to the K-means criterion, and using the `k-means++` algorithm (Arthur and Vassilvitskii, 2007).

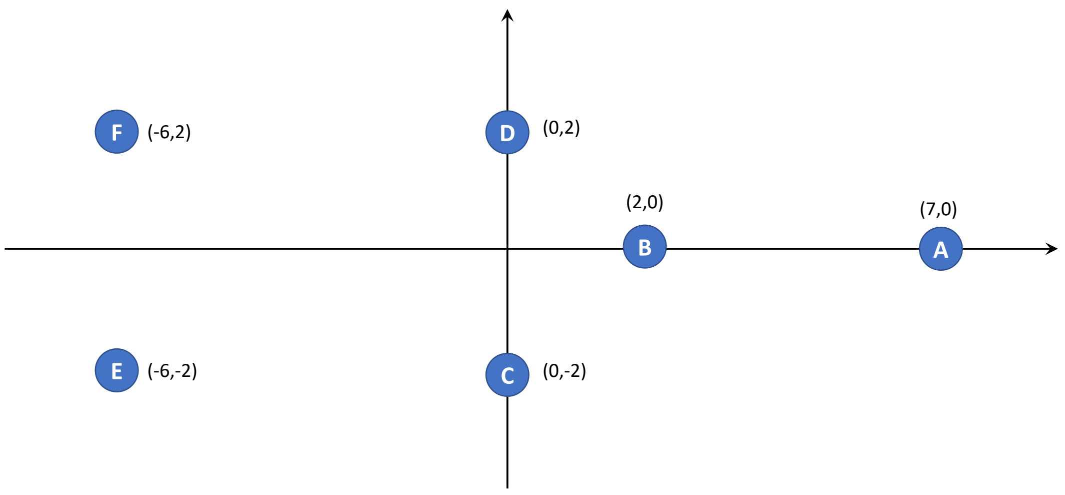

Figure 1: Dataset for Question 8

1. Consider $k = 1$. What is the cluster center? (I.e., what is the *centroid* of the entire dataset?)

Now consider $k = 3$. We would like to study how `k-means++` performs clustering on this dataset.

2. How does the algorithm choose the first initial cluster center c_1? (In other words, determine the probability of each data point being chosen as c_1).

3. Conditioned on A being chosen as c_1, how does the algorithm choose c_2?

4. Conditioned on $c_2 = F$ and $c_1 = A$, how does the algorithm choose c_3?

5. Suppose that the algorithm chooses $c_3 = B$ —hence, $c_1 = A, c_2 = F, c_3 = B$. Assume we stop here. Determine the clustering C based on the cluster centers c_1, c_2, c_3 and compute the corresponding cost ϕ.

   Recall that for the K-means problem, the cost ϕ of a dataset $\mathcal{X}$ and a set of cluster centers $\mathcal{C} = \{c_1, \ldots, c_k\}$ is

$$\phi(\mathcal{X}, \mathcal{C}) = \sum_{x \in \mathcal{X}} \min_{c \in \mathcal{C}} \|x - c\|_2^2$$

6. Now suppose we perform `k-means` using the initial cluster centers above (i.e., $c_1 = A, c_2 = F, c_3 = B$). Determine the output clustering C' and the corresponding cost ϕ.

   Report all necessary computations in Tasks 1–6.

References

David Arthur and Sergei Vassilvitskii. k-means++: The advantages of careful seeding. In *Symposium of Discrete Algorithms (SODA'07)*, pages 1027–1035, 2007.

R. J. Freund and P. D. Minton. *Regression Methods: A Tool for Data Analysis*, page 111. Marcel Dekker, 1979.

A. L. Goldberger, L. A. N. Amaral, L. Glass, J. M. Hausdorff, P. Ch. Ivanov, R. G. Mark, J. E. Mietus, G. B. Moody, C.-K. Peng, and H. E. Stanley. PhysioBank, PhysioToolkit, and PhysioNet: Components of a new research resource for complex physiologic signals. *Circulation*, 101(23):e215–e220, 2000. doi: 10.1161/01.CIR.101.23.e215.

C. Iber and AASM. *The AASM manual for the scoring of sleep and associated events: rules, terminology and technical specifications.* American Academy of Sleep Medicine, Westchester, I. L., 2007.

A. Kales and A. Rechtschaffen. *A manual of standardized terminology, techniques and scoring system for sleep stages of human subjects. Allan Rechtschaffen and Anthony Kales, editors.* U. S. National Institute of Neurological Diseases and Blindness, Neurological Information Network Bethesda, Md, 1968.

B. Kemp, A. H. Zwinderman, B. Tuk, H. A. C. Kamphuisen, and J. J. L. Oberye. Analysis of a sleep-dependent neuronal feedback loop: the slow-wave microcontinuity of the EEG. *IEEE Transactions on Biomedical Engineering*, 47(9): 1185–1194, 2000. doi: 10.1109/10.867928.

M. Perslev, M. Hejselbak Jensen, S. Darkner, P. J. Jennum, and C. Igel. U-time: A fully convolutional network for time series segmentation applied to sleep staging. In *Advances in Neural Information Processing Systems (NeurIPS)*, 2019.

Helmut Späth. *Mathematical Algorithms for Linear Regression*, page 305. Academic Press, 1991.

L. van der Maaten and G. Hinton. Visualizing data using t-SNE. *Journal of Machine Learning Research*, 9:2579–2605, 2008.