Machine Learning B

2024-2025

Home Assignment 3

Nirupam Gupta
Department of Computer Science
University of Copenhagen

The deadline for this assignment is **13 May 2025, 17:00**. 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.

In addition to submitting a pdf file containing response to the questions below, **please also submit your code as a separate zip-file.** In this assignment, you are free to use Python libraries however needed. Provide proper comments in your code, so it can be fairly evaluated.

1 SVM with Kernels (30 points)

Train the following binary SVM classifiers over the dataset provided in the "non-linear_svm_data.csv" file. Recall from the lectures that we denote by c the misclassification penalty and the Gaussian kernel is given by $K(x, x') = \exp\left(-\frac{\|x-x'\|^2}{2a^2}\right)$.

1. Linear SVM.
2. SVM with the Gaussian kernel.

Question 1. (15 points) Report your training loss for the Linear SVM with $r = 1$ and misclassification penalty $c = 1, 100, 1000$.

Question 2. (15 points) Report your training loss for SVM with Gaussian kernel with parameter $a = 0.1, 1, 10$.

(Optional) Try polynomial kernel with different values for parameter a (cf. lecture notes) and compare the performance with that of Gaussian kernel.

2 Logistic regression on MNIST (70 points)

In this exercise, you will design a *handwritten digit classifier* for digits $\{3, 8\}$. Specifically, you will be training a logistic regression model over images of digits 3 and 8 in the popular MNIST dataset. To get started on this task, you can refer to this online post that explains how to work with the MNIST dataset in python.

Train a logistic regression model with l_2-regularization parameter $\mu = 1$ (cf. lecture notes on optimization algorithms) using the following optimization algorithms over the images of digits labeled as $\{3, 8\}$ in the MNIST dataset.

1. Gradient descent method with a constant learning rate $\gamma = 0.001$.
2. Mini-batch stochastic gradient descent (mini-batch SGD) method with batch-size $b = 10$ and a constant learning rate $\gamma = 0.001$.

Question 3. (15 points) Run the different algorithms for at least 100 iterations and plot the respective training losses after every 10 iterations. You are free to run

more iterations (if the algorithm does not seem to converge after 100 iterations) or plot training losses with higher frequency.

Question 4. (20 points) Show and explain how the training losses for the different algorithms change with the learning rate γ. Report your findings for $\gamma = 0.0001, 0.001, 0.01$. However, feel free to play with additional learning rates as long as you properly specify them in your response.

Question 5. (20 points) How does changing the batch-size impact the convergence of mini-batch SGD? Report your findings for batch-size $b = 1, 10, 100$, with a constant learning rate γ that gives the best performance as per your findings in Question 4.

Question 6. (15 points) Re-run the mini-batch SGD method with batch-size $b = 10$ and a diminishing learning rate $\gamma_t = \frac{1}{t}$ where $t = 1, 2, \ldots$. Does it improve the performance over the best constant learning rate in Question 4?

(Optional) Explore what happens if instead of using a mini-batch stochastic gradient g_t, we use its momentum $m_t = \beta m_{t-1} + (1 - \beta) g_t$ where $\beta \in [0, 1)$ and $m_0 = 0$ to obtain w_{t+1}. That is, $w_{t+1} = w_t - \gamma m_t$. You can try different values for β to see how the convergence rate and the overall performance of the model changes.