Mining Massive Data (SS2025)

Pen & Paper 3

universität wien

Availability date: 24.04.2025
Exercise session: 08.05.2025 (attendance is mandatory)
Checkmark & upload date: 07.05.2025, 23:00

Number of tasks: 5
Maximum achievable points: 30 points

Submission & Grading

This pen & paper exercise is graded. You have to indicate which tasks you have solved and upload your solutions by the *checkmark & upload deadline* (see the header) on the course's Moodle webpage [2]. To be awarded the points for these tasks, you have to be present in the respective pen & paper session, be ready to present your solutions, and, if you are selected, provide a concise and comprehensible presentation of your solution which demonstrates your understanding of the problem & solution. The selection process will mainly be uniformly at random. **If you fail to be present in the session or are not able to explain your solution, you will not be awarded any points for the whole pen & paper session.**

Because of the large number of students enrolled in the course, we do not provide individual feedback on your solutions by default but we will provide sample solutions. If you want feedback on your solutions beyond the sample solutions, contact us via email (see the contact details below).

Questions

If you have any questions, please ask them in the respective Moodle forum. If you don't get an answer from your colleagues or us within 48 hours, reach out to us via email (ensure that you start the subject of your email with `[MMD25]`):

- Timo Klein (`timo.klein@univie.ac.at`)

Tasks

Task 1: Logistic Loss

(maximum achievable points: 4 points)

The support vector machine is not the only classifier with a convex loss function. For instance, there is also the so called *logistic regression* (LR). We consider a multi-class version of logistic regression in which $\mathbf{x} \in \mathbb{R}^d$ and $y \in \{1, \ldots, K\}$, i.e., the input space is d-dimensional and there are K different classes. The probability of predicting class y according to the LR model is

$$p(Y = y|\mathbf{x}) = \frac{\exp(\mathbf{w}_y^T \mathbf{x})}{\sum_{i=1}^{K} \exp(\mathbf{w}_i^T \mathbf{x})}, \tag{1}$$

where $\mathbf{w}_1, \ldots, \mathbf{w}_K \in \mathbb{R}^d$ are the parameters of the model. As a loss function, one typically considers the negative log-likelihood, i.e.,

$$\ell(\mathbf{x}, y; \mathbf{w}_1, \ldots, \mathbf{w}_K) = -\log(p(Y = y|\mathbf{x})). \tag{2}$$

(a) **[3 point(s)]** Prove that the negative log-likelihood is convex in $\mathbf{w}_1, \ldots, \mathbf{w}_K$. Don't use the fact that the log-sum-exp function is convex. But you can use the fact that functions linear in $\mathbf{w}_i$ are convex.

(b) **[1 point(s)]** Is the negative log-likelihood strongly convex? Prove or disprove.

Task 2: Support Vector Machine (SVM) in Online Convex Programming

(maximum achievable points: 5 points)

[5 point(s)] Implement an SVM in the online convex programming framework and report the regret

$$R_T = \sum_{t=1}^{T} \ell_t(\mathbf{w}_t) - \sum_{t=1}^{T} \ell_t(\mathbf{w}^*) \tag{3}$$

on the dataset `toydataset.csv` provided on Moodle [2], where

$$\mathbf{w}^* = \arg\min_{\mathbf{w} \in S} \sum_{t=1}^{T} \ell_t(\mathbf{w}), \tag{4}$$

$\ell_t(\cdot)$ is the hinge loss, and $S = \{\mathbf{w} \in \mathbb{R}^4 \mid \|\mathbf{w}\|_2 \leq 1\}$.

Report the regret for 5 different initial learning rates using the same ordering of the samples (you can choose it randomly). Make sure to anneal the learning rate as discussed in the lecture.

Furthermore, plot the instantaneous regret $\ell_t(\mathbf{w}_t) - \ell_t(\mathbf{w}^*)$ over t for 5 different learning rates. Use the same learning rates as before. What do you observe?

Task 3: Projection on the L1-norm

(maximum achievable points: 6 points)

Suppose we want to train a classifier or regression model with a 1-norm constraint on the parameter vector to encourage sparsity (and hence improve computation / interpretability). For using projected gradient descent like approaches, we need to compute a projection of the parameter vector $\mathbf{w} \in \mathbb{R}^d$ on the 1-norm ball, i.e.,

$$\mathbf{w}^P = \text{Proj}_S(\mathbf{w}) = \arg\min_{\mathbf{w}' \in S} \|\mathbf{w}' - \mathbf{w}\|_2, \tag{5}$$

where $\mathbf{w}^P$ is the projection of $\mathbf{w}$, and where $S = \{\mathbf{w}' \in \mathbb{R}^d \colon \|\mathbf{w}'\|_1 \leq 1\}$.

(a) **[1 point(s)]** Write down the Karush-Kuhn-Tucker conditions for the projection problem.

(b) **[4 point(s)]** What insights about the optimal solution can you get from the Karush-Kuhn-Tucker conditions? Devise an algorithm computing the projection (aim to exploit the insights you made).

(c) **[1 point(s)]** Consider a 2-dimensional problem. Compute the projection of the point $(2, 1)$ onto the ℓ_1 unit-ball using your devised algorithm. How does this projection compare to the projection of the same point onto the ℓ_2 unit-ball?

Hint: This problem is non-trivial to solve and does not have a nice closed-form solution. The idea of this exercise is to show that even simple looking projections might be difficult to compute and impose significant computational cost.

Task 4: Parallel Mini-Batch Gradient Descent

(maximum achievable points: 7 points)

Consider a convex optimization problem where we minimize a convex loss function

$$f(\mathbf{w}) = \mathbb{E}_{x\sim\mathcal{D}}[\ell(\mathbf{w};\mathbf{x})],$$

where $\mathbf{w} \in \mathbb{R}^d$ is the parameter vector and $\ell(\mathbf{w};\mathbf{x})$ is a convex loss function with respect to $\mathbf{w}$ for any data point $\mathbf{x}$ drawn i.i.d. from some distribution $\mathcal{D}$.

In stochastic gradient descent (SGD), at each iteration t, we sample a single data point $\mathbf{x}_t$ and update the parameter vector via:

$$\mathbf{w}_{t+1} = \mathbf{w}_t - \eta_t \nabla \ell(\mathbf{w}_t;\mathbf{x}_t).$$

To reduce variance and make use of parallel computation, we can instead draw a mini-batch of B independent samples $\{\mathbf{x}_t^{(1)},\ldots,\mathbf{x}_t^{(B)}\}$ at each step and compute the average gradient:

$$\mathbf{w}_{t+1} = \mathbf{w}_t - \eta_t \cdot \frac{1}{B}\sum_{i=1}^{B} \nabla \ell(\mathbf{w}_t;\mathbf{x}_t^{(i)}).$$

(a) **[4 point(s)]** Show that the variance of the mini-batch gradient estimator is reduced by a factor of $1/B$ compared to the variance of the single-sample estimator. To this end, let $g_i := \nabla\ell(\mathbf{w};\mathbf{x}^{(i)})$ and assume that the stochastic gradients g_i are i.i.d. with

$$\mathbb{E}[g_i] = \nabla f(\mathbf{w}), \quad \text{and} \quad \mathbb{E}\left[\|g_i - \nabla f(\mathbf{w})\|^2\right] = \sigma^2.$$

(b) **[3 point(s)]** Explain how this variance reduction impacts the convergence rate of SGD. Discuss under what conditions it is computationally beneficial to use large mini-batches in parallel implementations.

Hint: Note that a common convergence rate for SGD with step size $\eta_t = \frac{1}{\sqrt{t}}$ and bounded variance σ^2 is

$$\mathbb{E}[f(\bar{\mathbf{w}}_T)] - f(\mathbf{w}^*) \leq \mathcal{O}\left(\frac{\sigma}{\sqrt{T}}\right),$$

where $\bar{\mathbf{w}}_T$ is the average of iterates, and $\mathbf{w}^*$ is the optimal solution.

Task 5: Decision Trees

(maximum achievable points: 8 points)

[Reading exercise] Decision trees are commonly used classifiers and provide a human-readable explanation of how to derive decisions from them.

(a) **[4 point(s)]** Read and understand sections 12.5.1-12.5.5 of the MMDS book [1]. Prepare an explanation of what decision trees are, how they work, and how they are constructed. Of course, you can use all the material from the book.

(b) **[4 point(s)]** Decision trees can be computational demanding and using a single decision tree can lead to overfitting. Read and understand sections 12.5.6-12.5.8 of the MMDS book [1]. Be prepared to explain how to exploit parallelism in decision trees, how to reduce overfitting by node pruning, and what decision forest are.

Hint: It might be useful to prepare some sketches to support your explanation.

References

[1] Jure Leskovec, Anand Rajaraman, and Jeffrey David Ullman. *Mining of massive data sets*. Cambridge university press, 2020. URL: `http://www.mmds.org/`.

[2] *Mining Massive Data Moodle Page*. URL: `https://moodle.univie.ac.at/course/view.php?id=445688`.