Mining Massive Data (SS2025)

Pen & Paper 5

Availability date: 22.05.2025
Exercise session: 05.06.2025 (attendance is mandatory)
Checkmark & upload date: 04.06.2025, 23:00

Number of tasks: 4
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: Sequential k-means

(maximum achievable points: 8 points)

In this question you will consider an algorithm for sequential k-means clustering (see Algorithm 1). We would like to perform an assignment of examples $\mathbf{x}_1, \mathbf{x}_2, \ldots$ that are streamed, hence not all known a priori, into k clusters. One way to derive an algorithm is to perform online gradient descent on the k-means quantization error. We define the loss function $L(x, \mu)$ where μ is the vector of means $\mu_1, \ldots, \mu_k$ for each label, and $L(x, \mu) = \min_i \|\mathbf{x} - \mu_i\|_2^2$. Once each example arrives, the algorithm calculates the gradient $\partial L / \partial \mu$, and takes a step in the negative gradient direction. The gradient update then would be:

$$\frac{\partial L(\mathbf{x})}{\partial \mu_i} = \begin{cases} 2(\mu_i - \mathbf{x}) & \text{if } i = \arg\min_j \|\mathbf{x} - \mu_j\|_2 \\ 0 & \text{otherwise.} \end{cases}$$

Thus, for each new example $\mathbf{x}_t$, we first identify the currently closest cluster i with mean μ_i, and update it such that μ_i takes a step towards $\mathbf{x}_t$. An important design choice is the step size parameter (i.e., how quickly the mean should move towards the data points). In this question, you will analyze different choices.

(a) **[3 point(s)]** One approach is to use a fixed constant value $a \in (0, 1)$ to scale the step size towards the new point $\mathbf{x}$ (see Algorithm 1). This scaling constant a defines the influence of each new sample towards the new position of the closest mean. Let $\mathbf{x}'_1, \ldots, \mathbf{x}'_n$ be the first n examples assigned to cluster i (i.e., used to update μ_i). Let $\mu_i^{(n)}$ be the estimate of μ_i after it has been updated using n examples, and $\mu_i^{(0)}$ is the initial guess. Prove that after the arrival of the first n examples the means vector $\mu_i^{(n)}$ is equal to:

$$\mu_i^{(n)} = (1 - a)^n \mu_i^{(0)} + a \sum_{k=1}^{n} (1 - a)^{n-k} \mathbf{x}'_k$$

(b) **[3 point(s)]** Propose an alternative update rule such that $\mu_i^{(n)} = \frac{1}{n} \sum_{k=1}^{n} \mathbf{x}'_k$.

Hint: You might also have to change the initialization of the clusters.

(c) **[2 point(s)]** Which of the two update rules would you prefer in which situation?

[Credit: Taken from Andreas Krause's Data Mining course.]

Algorithm 1 Sequential k-means

```
1: Initialize $\mu$ with an initial guess $\mu_i^{(0)}$ for each $\mu_i$
2: for $t = 1, 2, \ldots$ do
3:     $i \leftarrow \arg\min_j \|\mathbf{x}_t - \mu_j\|_2^2$ (find the closest cluster $i$)
4:     $\mu_i \leftarrow \mu_i + a(\mathbf{x}_t - \mu_i)$ (take step towards data point)
5: end for
6: Return $\mu$ (return the means of all $k$ clusters)
```

Task 2: Actively Learning a Union of Intervals

(maximum achievable points: 8 points)

Suppose you are given a pool $X = \{x_1, \ldots, x_n\}$ of n unlabeled examples where each $x_i \in [0, 1]$. Further suppose there are unknown constants $0 \leq a < b < c < d \leq 1$ such that all $x_i \in [a, b] \cup [c, d]$ are labeled with 1, whereas all remaining points are labeled with -1. The goal of this task is to develop a pool-based active learning scheme that infers the labels of all unlabeled examples. The algorithm should sequentially select one of the n examples and obtain its true label (i.e., there is no noise).

(a) **[2 point(s)]** Show that in general n labels are needed to infer the labels of all unlabeled examples.

(b) **[3 point(s)]** For $x < x'$ define $E(x, x') = |X \cap [x, x']|$, i.e, the number of examples contained in the interval $[x, x']$. Suppose $E(a, b) \geq m$, $E(b, c) \geq m$ and $E(c, d) \geq m$ for some known constant $m \geq 1$. Define an active learning scheme that selects examples given knowledge of m. How many samples are needed as a function of m and n?

Hint: Think of a query strategy exploiting what you know about the structure of the problem, i.e., m.

(c) **[3 point(s)]** Come up with an algorithm that works even without knowledge of m. That is, develop an algorithm that uses (approximately) the same number of labels as the algorithm from the previous subtask with $m = \min\{E(a, b), E(b, c), E(c, d)\}$? You can use a randomized algorithm and bound the expected number of labels requested.

Hint: This task is non-trivial to solve. Ask any upcoming questions early on.

[Credit: Taken from Andreas Krause's Data Mining course.]

Task 3: UCB

(maximum achievable points: 8 points)

In this exercise, we will prove a regret bound of the UCB1 algorithm. We will make the simplifying assumption that we know the total number of rounds T beforehand. We assume that there are k arms with random payoffs in $[0, 1]$ and means $\mu_1, \dots, \mu_k$, and that the optimal mean is $\mu^* = \max_{i=1}^k \mu_i$. Furthermore, let $\hat{\mu}_i^t$ be the empirical estimate of the mean μ_i at time t and denote by $\Delta_i = \mu^* - \mu_i$ the *suboptimality-gaps*. The full algorithm is given in Algorithm 2

Algorithm 2 UCB1 Policy for k-armed bandits with fixed T

```
1: Initialize $\hat{\mu}_i^0 = 0, n_i^0 = 0$ for all $i \in \{1, \dots, k\}$
2: Play each arm once for initialization and update $\hat{\mu}_i^t$ and $n_i^t$ accordingly
3: for $t = k+1, \dots, T$ do
4:     Pick arm $j \leftarrow \arg\max_i \hat{\mu}_i^t + \sqrt{\log(T)/n_i^t}$
5:     Update count $n_j^{t+1} \leftarrow n_j^t + 1$ and the mean estimate $\hat{\mu}_j^{t+1} \leftarrow \hat{\mu}_j^t + \frac{y^t - \hat{\mu}_j^t}{n_j^t + 1}$
6: end for
```

We will prove that the expected regret $\mathbb{E}[R_T] = T\mu^* - \mathbb{E}[\sum_{t=1}^T y_t]$ of UCB1 after T rounds is at most

$$\mathbb{E}[R_T] \le 4 \sum_{\Delta_i > 0} \frac{\log(T)}{\Delta_i} + 5 \sum_{\Delta_i > 0} \Delta_i = \mathcal{O}\left(\frac{k \log(T)}{\min_i \Delta_i}\right)$$

(a) **[1 point(s)]** Denote by n_i^t the number of times arm i has been played until round t (note that this is a random variable). Show that the total expected regret can be written was $\mathbb{E}[R_T] = \sum_{i=1}^k \mathbb{E}[n_i^T]\Delta_i$.

Hint: Use the so-called tower-rule.

(b) **[2 point(s)]** Next, we define a confidence set $\mathcal{C}_i^t = \{\mu : |\mu - \hat{\mu}_i^t| \le \sqrt{\frac{\log(T)}{n_i^t}}\}$ for each arm i. Note that the UCB1 policy plays the arm with the largest upper bound of the confidence set. Use Hoeffding's inequality to show that

$$P(\mu_i \notin \mathcal{C}_i^t) \le \frac{2}{T^2} \qquad \text{for any } t = 1, \dots, T.$$

(c) **[2 point(s)]** Let i^* denote the index of an optimal arm, i.e., $\mu_{i^*} = \mu^*$. Consider any suboptimal arm i and show that if $\mu_i \in \mathcal{C}_i^t$ and $\mu_i^* \in \mathcal{C}_{i^*}^t$ for all $t = 1, \dots, T$, then $n_i^T \le \frac{4\log(T)}{\Delta_i^2} + 1$.

(d) **[2 point(s)]** Use the probabilistic bounds above to bound the expected number of times $\mathbb{E}[n_i^T]$ a suboptimal arm i played, and put everything together to obtain the desired regret bound.

The bound we derived in the first part of the exercise is called an instance dependent regret bound as it contains the sub-optimality gaps Δ_i. In particular the bound degrades as $\Delta_i \to 0$.

(e) **[1 point(s)]** Use the regret decomposition and that $\mathbb{E}[n_i^T] \in \mathcal{O}(\frac{\log(T)}{\Delta^2})$ to prove the worst-case regret bound $\mathbb{E}[R_T] = \mathcal{O}(\sqrt{kT\log(T)})$.

[Credit: Taken from Andreas Krause's Data Mining course.]

Task 4: Clustering Streaming Data

(maximum achievable points: 6 points)

[Reading exercise] In this exercise we will read more about clustering data streams.

(a) **[3 point(s)]** As background, read and understand sections 4.1 and 4.6 of the MMDS book [1]. Prepare an explanation of the described streaming model and the presented approach for counting ones in data streams. Of course, you can use all the material from the book.

(b) **[3 point(s)]** Read and understand section 7.6 of the MMDS book [1]. Prepare an explanation of the described "BDMO algorithm" and how parallelism can be used for clustering. Of course, you can use all the material from the book.

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.