Mining Massive Data (SS2024)

Programming Assignment 3

Scalable Clustering Techniques

Availability date: 06.05.2024
Submission due date: 17.06.2024
Peer-review due date: 24.06.2024

Number of tasks: 1
Maximum achievable points: 100 points (60+30+10 ... Submission / Review / Quality of review)

General Remarks

- This is one of 3 programming assignments. For each assignment you can earn up to 100 points. Details on grading are provided below.
- The deadline for this assignment is noted on the header. No deadline extensions will be granted. Upload your solution by the deadline on the course's Moodle page [4]. If you fail to upload your solution by the deadline you will not be awarded any points.
- If you have questions or encounter problems, do not hesitate to contact us.
- Solve this programming assignment in Python. Don't use any non-standard libraries in your code, e.g., only use `numpy`, `scipy`, `scikit-learn`, unless explicitly mentioned.
- Make sure that your code is executable, e.g., if you created a jupyter notebook, make sure that the notebook can be executed without error when you reset the kernel and execute the notebook from the beginning. **If your code is not executable, we reserve the right to not award any points.**
- Solve the tasks on your own (considered group-wise). If we observe plagiarism or group work, we reserve the right to not award any points.
- Pack your code, your report, and execution instructions into a single .zip file with the following name: `group(group number).zip` file (e.g., `group01.zip`) and upload it on Moodle. Make sure that your code is executable, e.g., if you created a jupyter notebook, make sure that the notebook can be executed without error when you reset the kernel and execute the notebook from the beginning.
- Only **one** team-member submits the zip file in the Moodle system.

Questions

Please ask questions primarily on the respective Moodle forum. If you don't get a response in reasonable time (48 hours) from us or your colleagues, get in touch with our tutor via mail (Megi Teneqexhi). If you don't get a response in reasonable time (48 hours) from the tutors, get in touch with the rest of us (Simon Rittel, Timo Klein, Sebastian Tschiatschek).

Improving the Performance of k-Means Clustering using LSH and Coresets

Goal. The goal of this assignment is to study different approaches for approximately solving the k-means problem. In particular, you will (a) implement Lloyd's algorithm as a baseline algorithm for k-means clustering, and improve its runtime performance using (b) Locality Sensitive Hashing (LSH) and (c) coresets.

Regarding different variants of k-means clustering you will study the impact on accuracy and runtime performance of all three variants.

Data The KDD Cup was an annual data mining competition organized by a special interest group of the ACM. Each year, a machine learning problem was posed, along with a data set, and researchers were invited to submit papers describing their best solutions to the problem. Here we are using the KDD dataset from 2004, where they targeted a protein matching problem with this dataset. You can find some more details to the dataset here:

`https://www.kdd.org/kdd-cup/view/kdd-cup-2004/Data`

We will use this dataset for k-means clustering and some performance enhancements using LSH and coresets. We provide a `.csv` file in Moodle for download. The first element of each line is a `BLOCK ID` that denotes to which native sequence this example belongs. There is a unique `BLOCK ID` for each native sequence. `BLOCK ID`s are integers running from 1 to 303 (one for each native sequence, i.e., for each query). `BLOCK ID`s were assigned before the blocks were split into the train and test sets, so they do not run consecutively in either file. We will use this first column to evaluate our clustering result. There are 153 distinct values, therefore, we consider the true number of clusters with $k = 153$.

Evaluating clustering accuracy Evaluate your clustering result based on the normalized mututal information (NMI) measure with this column. The NMI measures how accurately the identified cluster assignments align with the true clusters of the samples. The NMI ranges between 0 and 1. The closer the value to 1, the better is your clustering. You can use sklearn's function `sklearn.metrics.normalized_mutual_info_score` for computing the NMI (set `average_method='arithmetic'`).

Assignment

Lloyd's algorithm for k-Means Clustering (34%). As a baseline model implement Lloyd's algorithm [3] for k-means clustering and initialize it with the first k points as initial cluster centers. The default convergence criteria is to stop the algorithm if none of the cluster memberships have changed in comparison to the previous iteration.

☐ Include a plot illustrating convergence of k-means.

☐ Track the number of iterations needed for convergence and compare it to the other implementations.

☐ Report the achieved NMI averaged over at least 5 runs.

☐ Report the runtime in [sec] for your algorithm averaged over at least 5 runs. Also report the number of distance computations performed.

☐ Briefly discuss your implementation of Lloyds algorithm.

k-Means with Locality Sensitive Hashing (LSH) (33%) Implement Lloyd's algorithm using LSH to speed up the distance calculations. See the uploaded presentation on Moodle on how LSH should be used.

Try out different settings in which you combine different hash functions with AND and OR as it was discussed in the lecture. For example, you can have one setting where you combine two hash functions with AND and in the second setting you combine two functions with OR, but you could try out several combinations, with different hash functions. Also try varying the number of buckets of your hashing function. Measure the runtime $time_{LSH}$ of your LSH implementation.

☐ Report how you selected the parameters of LSH and how you combined your functions.

☐ Report the accuracy using NMI and the runtime in seconds averaged over at least 5 runs. Also report the number of distance computations performed. If your implementation doesn't show a speed-up, discuss why this might be and also discuss whether this situation would change when working larger datasets.

☐ Track the number of iterations needed for convergence (if it converges at all) and compare it to the other implementations.

☐ Briefly discuss your implementation of k-means with LSH.

k-means with coresets (33%). Coresets are a compact representation of data sets, such that models trained on a coreset are competitive with models trained on the full dataset [2, 1]. In this task you will implement coresets for k-means clustering as in [1, Algorithm 1]. For the the number of samples m, use 100, 1000, and 10000.

☐ Report the runtime and NMI you achieve when using coresets of different size (as described above) averaged over at least 5 runs. To do so, cluster the coresets using sklearn's k-means algorithm (you can supply sample weights to all needed functions).

☐ Track the number of iterations needed for convergence and compare it to the other implementations.

☐ Analyze the variance of the accuracy obtained when using coresets for clustering by computing 10 coresets for each choice of m.

Report

Write a report about your work on this assignment, your findings and results. Make sure to report all the information indicated above. Additionally report the following:

☐ Show the performance in terms of NMI and runtime for the different approaches in one plot or table.

References

[1] Olivier Bachem, Mario Lucic, and Andreas Krause. "Scalable k-Means Clustering via Lightweight Coresets". In: *Proceedings of the 24th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining, KDD 2018, London, UK, August 19-23, 2018*. ACM, 2018, pp. 1119–1127. DOI: 10.1145/3219819.3219973. URL: https://doi.org/10.1145/3219819.3219973.

[2] Sariel Har-Peled and Soham Mazumdar. "On coresets for k-means and k-median clustering". In: *Proceedings of the 36th Annual ACM Symposium on Theory of Computing, Chicago, IL, USA, June 13-16, 2004*. ACM, 2004, pp. 291–300. DOI: 10.1145/1007352.1007400. URL: https://doi.org/10.1145/1007352.1007400.

[3] Stuart P. Lloyd. "Least squares quantization in PCM". In: *IEEE Trans. Inf. Theory* 28.2 (1982), pp. 129–136. DOI: 10.1109/TIT.1982.1056489. URL: https://doi.org/10.1109/TIT.1982.1056489.

[4] *Mining Massive Data Moodle Page*. URL: https://moodle.univie.ac.at/course/view.php?id=428454.