

Part I «Portfolio-Exam»

MADS-EMDM

This is the first part of the portfolio-exam for the Data Science course MADS-EMDM (Advanced Topics of Data Mining), worth 49% of the final grade (98 points).

This exam is homework. Student's are allowed to exchange ideas. However, this is NOT a teamwork exercise. Every student must derive and write up their own solutions in their own words and programming style. To complete this first part of the exam,

- solve ALL the following tasks (4 pages!),
- create a Jupyter Notebook for your (commented) code as well as all textual answers (English or German), and
- upload both files to Moodle **before 23:59 o'clock (German time) October 25, 2022**.

Rules and Hints:

- In your notebooks, please complete the tasks in sequence.
- I will rerun the notebook. Make sure, everything is in the correct order and self-contained.
- Whenever random functions are used, set their seed to 1 (unless stated otherwise) to make the experiments reproducible.
- You are free to reuse code from the lectures and exercises.
- It is well possible, that you will have to look up certain notions before you can answer a specific question. This is intended! Try to find reliable, valid sources.
- Try to adhere to the DRY principle (Don't repeat yourself!). Use parametrized functions instead of code copy for repetitive tasks.
- **Do NOT FORGET to add textual answers to EACH task. Code alone is not sufficient!**

Exercise 0. (Self-Study. 0 Points)

Familiarize yourself with the Python library `timeit` as well as with the Jupyter line magic commands `%time` and `%timeit`.

Exercise 1. (Candidate Generation. 12 Points)

Consider the following set of items: a, b, c, d, e, g, m in some transaction database. Assume that APRIORI is running and has already computed the set L_4 using alphabetical order on the items. The resulting set is:

$$L_4 = \{ \begin{aligned} &\{a, b, c, d\}, \{a, b, c, e\}, \{a, b, c, m\}, \{a, b, d, e\}, \{a, b, e, m\}, \\ &\{a, c, e, m\}, \{a, d, g, m\}, \{b, c, d, e\}, \{b, c, d, g\}, \{b, c, d, m\}, \\ &\{b, c, e, g\}, \{b, c, e, m\}, \{b, d, e, g\}, \{c, d, e, g\}, \{c, d, g, m\} \end{aligned} \}$$


1. (10 points) Use the APRIORI candidate generation (and alphabetical order on the items) to create the set C_5 of candidates for 5-element frequent itemsets. Explain your decisions to create or discard elements.
2. (2 points) For how many itemsets does the database have to be scanned to yield L_5 from C_5 ?

Notes for the following tasks:

- Throughout the following tasks, you will use two transaction datasets of different size to conduct various experiments.
- For the implementations of APRIORI and FP-Growth use the module `mlexend`.
- The following experiments are time consuming (the full notebook will probably take several hours to finish). Before running the experiments, think of useful strategies for testing your code.

Exercise 2. (Initial Data Analysis. 10 Points)

Load and briefly analyze two datasets. For that purpose conduct the following steps:

1. (2 points) Create the following grocery dataset:
 - a) fish, apples, cider, dragon fruit, garlic, ice cream, mints, prunes
 - b) apples, bacon, cider, fish, lemons, mints, oatmeal
 - c) bacon, fish, ham, jam, oatmeal
 - d) bacon, cider, kiwis, spam, prunes
 - e) apples, fish, cider, eggs, lemons, prunes, mints, nachos
2. (5 points) Load the dataset T10I4D100K from the  Frequent Itemset Mining Dataset Repository. For the purpose of these experiments, use only the first 10,000 transaction of this database (as ordered in the `.dat` file).
3. (3 points) For each dataset compute the highest (relative) support that a non-empty itemset actually reaches.

Exercise 3. (A first Impression on Runtime. 16 Points)

Get a first impression on the runtime of APRIORI on the two datasets in their non-sparse representation:

1. (3 points) Run and time APRIORI once on the grocery dataset using a support threshold of 0.4 and a fitting line magic command. How long did the run take? How many frequent itemsets were found?
2. (3 points) Run and time APRIORI once on the T10I4D100K dataset using a support threshold of 0.004 and a fitting line magic command. How long did the run take? How many frequent itemsets were found?
3. (4 points) Time repeated runs of APRIORI on both datasets using the above respective support thresholds and a fitting line magic command where
 - on the grocery dataset, 10 measurements are taken, each relying on 10 executions and
 - on the T10I4D100K dataset, 10 measurements are taken, each relying on 1 execution.

Report average runtime together with standard deviation for both experiments.

4. (6 points) Discuss limitations of the above approaches!

Exercise 4. (The Influence of Sparse Data Representations. 28 Points)

Compare the runtimes of the APRIORI algorithm on the two datasets both in sparse and non-sparse representations. For this task, run APRIORI without column names.

1. (15 points) Time and run APRIORI on both datasets, first on the regular (non-sparse) representation, then on a sparse representation.
 - For the grocery dataset use 0.2, 0.4, 0.6, 0.8 as the support thresholds, run 10 measurements for each setting and use 10 executions for each measurement.
 - For the T10I4D100K dataset use 0.002, 0.004, 0.008, 0.016, 0.032, 0.064 as the support thresholds, run 10 measurements for each setting and use 1 execution for each repetition.

Out of the 10 measurements, report only the minimum for each setting (column parameter and support threshold).

2. (5 points) Display results on each dataset in a separate, suitable table (create useful columns), focussing on the comparison of runtime and memory consumption for executions on non-sparse and sparse data structures.
3. (5 points) Interpret the results – state observations, limitations, advice regarding the use of sparse representations!
4. (3 points) Explain, why only the minimum runtime is reported (in contrast to other options, like the average runtime).

Exercise 5. (APRIORI vs. FP-Growth. 32 Points)

Compare the runtime of the APRIORI algorithm to those of FP-Growth on the two datasets. For this task, run the algorithms without column names on the non-sparse data structures.

1. (15 points) Time and run APRIORI and FP-Growth on both datasets.
 - For the grocery dataset use 0.2, 0.4, 0.6, 0.8 as the support thresholds, run 10 measurements for each setting and use 10 executions for each repetition, and
 - for the T10I4D100K dataset use 0.002, 0.004, 0.008, 0.016, 0.032, 0.064 as the support thresholds, run 10 measurements for each setting and use 1 execution for each repetition.

Out of the 10 measurements, report only the minimum for each setting (column parameter and support threshold).

2. (5 points) Display results on each dataset in a separate, suitable table (create useful columns), focussing on the comparison of runtime for executions of the two algorithms.
3. (7 points) Add a visual comparison by plotting the runtimes of both algorithms against the support thresholds into the same diagram (thus, one diagram per dataset). Use scaled axes where useful.
4. (5 points) Interpret the results – state observations, limitations, advice regarding the use of the two algorithms!