University of Vienna
Faculty of Computer Science
Prof. Claudia Plant
MSc. Ylli Sadikaj
BSc. Rona Perjuci

Foundations of Data Analysis : Lab Exercise A2

SoSe 2022

General Remarks:

- The deadline for the submission is at 09:45 a.m. on 23.06.2022. Please upload your solutions on Moodle. No deadline extension is possible.
- After the above deadline, the peer review process on Moodle will start and you have to evaluate two other students' work until 30.06.2022 at 11:59 p.m., this means your own work will be reviewed as well. You can find all the further details about the peer review process on Moodle.
- For answering the questions, please use Python. Your code must be well-documented and readable; see the Style Guide for Python[1]for common style conventions.
- Upload your solutions as a zip archive with the following naming scheme **matrikelnumber_A2.zip**. The archive should contain your code as well as a PDF report (or HTML file if you work with jupyter notebooks) with your assumptions, results and a description how to compile and run your code.
- Mark and cite external sources you are using in the code and report.
- If you have problems do not hesitate to contact the tutors or post a question on the dedicated Moodle forum.

Task -1 Apriori Algorithm for Recommender Systems (40P)

Task Definition:
In this programming assignment, you are required to implement the Apriori algorithm and apply it to mine frequent itemsets for book recommendation. You are required to implement the algorithm from scratch by using only native Python libraries and numpy.

Input: The provided input file (books.txt) is based on the Book-Crossing dataset[1]. It contains the favourite books of 1850 users. Each line in the file corresponds to a user and represents a list of books the user likes. An example:

The Da Vinci Code;Jurassic Park

In the example above, the corresponding user likes the books "The Da Vinci Code" and "Jurassic Park".

Output: You need to implement the Apriori algorithm and use it to mine sets of books that are frequent in the input data. After implementing the Apriori algorithm, please set the relative minimum support to 0.01 and run it on the 1850 lists of books. In other words, you need to extract all the itemsets that have an absolute support larger than 18

[1]`https://www.python.org/dev/peps/pep-0008/`

(a) Output all the length-1 frequent books with their absolute supports into a text file named "oneItems.txt" and place it in the root of your zip file. Every line corresponds to exactly one frequent book and should be in the following format:

Support:book

For example, suppose a book (e.g. The Da Vinci Code) has an absolute support 68, then the line corresponding to this frequent item set in "oneItems.txt" should be:

68:The Da Vinci Code

(b) Please write all the frequent itemsets along with their absolute supports into a text file named "patterns.txt" and place it in the root of your zip file. Every line corresponds to exactly one frequent itemset and should be in the following format:

support:book_1;book_2;book_3;...

For example, suppose an itemset (The Da Vinci Code;The Secret Life of Bees) has an absolute support 20, then the line corresponding to this frequent itemset in "patterns.txt" should be:

20:The Da Vinci Code;The Secret Life of Bees

(c) Imagine you should recommend a book to a user. You know that the user likes the books "Harry Potter and the Sorcerers Stone (Book 1)" and "Harry Potter and the Chamber of Secrets (Book 2)". Based on the result of the Apriori algorithm, give a book recommendation for this user by maximizing the confidence that the user will like the book. Explain your choice and report the confidence score for your recommendation.

Task -2 Dimensionality Reduction, Latent Semantic Indexing (30 P)

Latent semantic indexing (LSI) is an indexing and retrieval method that uses the singular value decomposition to identify patterns in the relationships between the terms and concepts contained in unstructured text. In this exercise it is highly recommended to use existing libraries such as `sklearn` and `gensim`[2]. We also provide a python template `lsi_template.py` for the preprocessing of text. This has the advantage, that your results will be comparable to others.

(a) Use the `20newsgroups` dataset and use gensim to apply the LSI transformation on `tf-idf` model. Make yourself familiar with the `tf-idf` measure (10 points).

(b) Pick a random document and try to query it with the terms contained in the document. Summarize your findings and interpret the result in the report (20 points).

Task -3 Clustering (30 P)

In this task you try different clustering algorithms and evaluate them using Normalized Mutual Information (NMI) and the (Adjusted) Rand Score. Use four different datasets: *dataset1_noClusters2.csv*, *dataset2_noClusters2.csv*, *dataset3_noClusters2.csv*, and *dataset4_noClusters7.csv*.

(a) Import and plot the data and if necessary preprocess it using *sklearn.preprocessing*.

(b) Try four different clustering techniques such as DBSCAN, KMeans, Expectation Maximization (EM), and Average Link, which are already implemented in *scikit-learn*. Please state which algorithms you have used and what parameters you have chosen and why you have chosen them.

[2]gensim URL: `https://radimrehurek.com/gensim/`

(c) Evaluate each clustering technique using Normalized Mutual Information[3] as well as the (Adjusted) Rand Score. For the evaluation, use the *sklearn.metrics* package.

(d) Briefly discuss your results.

References

[1] Ziegler, C.N., McNee, S.M., Konstan, J.A., Lausen, G.: Improving recommendation lists through topic diversification. In: Proceedings of the 14th International Conference on World Wide Web. p. 22–32. WWW '05, Association for Computing Machinery, New York, NY, USA (2005). https://doi.org/10.1145/1060745.1060754, `https://doi.org/10.1145/1060745.1060754`

[3] `https://nlp.stanford.edu/IR-book/html/htmledition/evaluation-of-clustering-1.html`