# CS-E4650 Methods of Data mining

## Exercise 5 / Autumn 2023

### 5.1 Social network among school pupils

*Learning goal: Basic analysis of social networks (centrality measures, communities and their interpretation)*

In this task, you should experiment with the Gephi tool `https://gephi.org/` and analyze a social network among school children. The network describes interaction among class 5 primary school pupils during one day. The data is presented as two spreadsheets, **nodesclass5.csv** (nodes and their attributes) and **edgesclass5.csv** (edges and their weights). Each actor has two attributes: school class (5A or 5B) and gender. The edge weights weight reflect the strength of interaction (total duration).

Install Gephi and start it. Then import the data: From the `file` menu, choose `import spreadsheet` and choose nodesclass5.csv data. The edges are loaded similarly, but in the end tick `append to existing workspace`. Now you should have under `Table` both Nodes and Edges. You can find instructions how to use Gephi on `https://gephi.org/users/quick-start/` and `https://github.com/gephi/gephi/wiki/`. The functions are available under `statistics` and `data table` shows values of nodes for all calculated measures. After running a function, you can also visualize results under `appearance` → `ranking`.

a) Identify the most central and influential nodes with the following measures. Report at least two nodes with each measure (more if many nodes with the same value) and explain what the high value means.

   i) Node degree (select network overview – average degree)

   ii) Weighted degree (network overview – average weighted degree)

   iii) Closeness centrality (network overview – network diameter)

   iv) Betweenness centrality (already calculated in iii)

   v) Why the top-two nodes with highest weighted degree are not among top degree nodes? (Hint: look at their edge weights.)

   vi) Which are the most critical nodes for the information flow?

b) Identify dense subgraphs (communities) with the Modularity function (select network overview – modularity). What kind of communities do

you find? Try to explain the communities with background variables (class and gender)!

c) Make some small experiment of your own choice with gephi and report the results! You can e.g., visualize some aspects of the network, calculate more measures or test Newman-Girvan community detection (first install the plugin: under `tools` → `plugins` choose `Newman-Girvan clustering`).

## 5.2 Sequential data of discrete events

*Learning goal: How to test significance of discoveries with randomization.*

Consider sequential data where discrete events of three types, $A$, $B$ and $C$ occur in time. That is, the data consists of tuples $(D, t)$, where $D$ is one of $A$, $B$ and $C$ and $t$ is the occurrence time.

An example sequence $S$ could be:

$$\{(A, 105), (B, 110), (A, 120), (C, 122), (A, 130), (B, 135), (C, 185), (A, 195), (C, 220),$$
$$(A, 260), (B, 270), (C, 295), (A, 420), (C, 440), (B, 445), (C, 522), (A, 530), (B, 555)\}.$$

These events could be, e.g., different types of problems arising in a piece of machinery. Browsing the data you notice that it looks like many occurrences of $A$ are followed fairly soon by an occurrence of $B$.

Questions:

a) How would you quantify this possible connection between occurrences of $A$ and $B$? Explain what you would compute from the data and show some values with the given example sequence $S$.

b) How would you use randomization to test whether the value you computed is in some sense interesting?

Please be specific wrt. your assumptions about the sampling distribution you would use, e.g., are you keeping something constant or letting it vary (how and why). What assumptions are you making about the data?

## 5.3 Preprocessing NLP data

*Learning goals: preprocessing text data, detecting errors and special cases; how to increase frequency of important terms and collocations despite different spelling and grammatic forms*

In this task, we will practise preprocessing of text data with Python Natural Language Toolkit (nltk). The idea is that you can utilize the results in the homework task, even if you were using another programming language for the actual processing.

Install packages nltk, scikit-learn and numpy with command `pip3 install scikit-learn nltk numpy`. There is also a book "Natural Language Processing with Python" `https://www.nltk.org/book/` with examples.

Load example data **acmdocuments.txt** from MyCourses. Each line is considered as one document. They are sentences from scientific abstracts in the ACM digital library `https://dl.acm.org/`. In MyCourses, you can find a code skeleton **preprocSbS.py** that you can use as a starting point, unless you already know how to do all preprocessing steps.

a) The main tasks of preprocessing text data are tokenization, lowercasing, removing punctuation, stemming (or lemmatization), and stop-word removal. However, order of these steps depends on the library and your implementation. The stopword list may contain only stopwords in their normal (unreduced) form, in a stemmed form, or with punctuation characters like (apostrophes). Lemmatization tools often require full sentences so that they can utilize part-of-speech analysis. The first task is always to determine the right order to do the preprocessing steps. Make a fast sanity check to the example code: is it performing the steps as desired? What would happen if you skipped lowercasing or performed stemming before stopword removal?

b) Check stopword removal and search examples of two types of errors: i) Stopwords (or other common and useless words in this context) that remain in the text, and ii) important words that are removed as stopwords (Hint: look important computer science abbreviations and notations in the NLTK stopword list). Estimate how serious these errors are (assuming we had a larger corpus of similar documents). How could you fix the (most serious) errors?

c) Check the quality of stemming (with Porter stemmer). Can you find errors where either i) two words having the same basic forms are reduced to different stems or ii) two words with different roots are reduced to the same stem? Test if the Snowball stemmer would do a better job! Are there errors where lemmatization could help?

d) Check punctuation removal. Can you find errors where either i) punctuation that should be removed has remained or ii) punctuation that

is important for the meaning of the term has been removed? No need to check hyphenated words, yet.

e) Evaluate collocations/compound words (phrases of multiple consecutive words). Can you find examples of important collocations that occur in different forms: i) closed (constituent words catenated together), ii) hyphenated (hyphen between words), ii) open (space between words). Suggest a solution how to handle them!

## 5.4 Homework: Topics of text clusters

*Learning goal: Clustering text data and techniques for describing topics of clusters*

In this task, you should cluster a collection of short scientific texts and identify the main topics of each cluster. Ideally, you will indentify 3–10 unique topics (areas or techniques of computer science) that describe majority of documents excluding possible outliers.

In MC, you can find data set **scopusabstracts.csv**, which consists of abstracts of scientific papers from Scopus `https://www.elsevier.com/products/scopus`. Each line describes one document: its id, title, and abtract, separated by #.

a) In the baseline solution, combine the title and abstract. Preprocess the data like in the previous task, but this time, create also **bigrams** (in addition to unigrams) as possible features. Since the number of features would otherwise be too high, it is suggested to use frequency-based filtering to prune out very frequent or extremely rare words/collocations (see parameters of sklearn TfidfVectorizer). Consider also adding new stopwords, if any frequent but uninformative words complicate later steps. When features are fine, present the data in the tf-idf form so that each document vector is normalized to unit $L_2$ norm.

b) Cluster the preprocessed data with $K$-means trying $K = 3, \ldots, 10$. Evaluate the clustering quality with the Davies-Bouldin index and select the best $K$. Then evaluate the most frequent unigrams and most frequent bigrams in each cluster. (It is possible that the lists still contain some uninformative stopwords that you need to exclude.) Try to conclude what is the topic of each cluster. This is the baseline solution, so don't worry, if all the topics are not yet clear.

c) Try to improve your results! Here you can freely try any methods covered in the course. You can improve the preprocessing (e.g., lemmatization), clustering (e.g., try dimension reduction or another clustering method) or the evaluation of the most important terms (e.g., utilize the title, perform SVD per cluster and look at the leading singular vector or analyze only the centroid or most central documents). Conclude the main (3–10) topics of the document collection based on your experiments!

**Parts of the report:**

1. Cover page: title (course name and assignment number), names and student ids of all participants of the team.

2. Section 1 "Preprocessing": Describe briefly the preprocessing methods: tools (like nltk), in which order the steps were performed, stemmer, stopword list (including own additions), tf-idf version (equation), minimum or maximum frequencies (if any), and other possible steps or options that could affect the results.

3. Section 2: "K-means clustering and topic detection": Report here the results of the $K$-means approach. What was the best clustering ($K$ and Davies-Bouldin index), the most frequent unigrams and bigrams in clusters (e.g., in a table), and your conclusion on the topics.

4. Section 3: "Additional experiments": Report here your experiments in the c) part. Describe briefly what you tried and the results (the most important terms and concluded topics). Evaluate also if your experiment was successful, i.e., if it produced better results than the baseline. It is suggested to divide this section into subsections, if you tried many approaches.

5. Section "Appendix": Include here the code of your program.

**Produce a pdf report including all parts and submit it in MyCourses before the deadline. Recall that this task is homework that is done in groups of 2–3 students – you cannot do the task alone or in a larger group.** You can search collaborators in zulip, exercise sessions, or ask help from the TAs.