OSTBAYERISCHE
TECHNISCHE HOCHSCHULE
REGENSBURG

INFORMATIK UND
MATHEMATIK

Natural Language Processing

Thema:	Exercise sheet 1: Text analysis and processing on the command line
Von:	Timo Baumann, `timo.baumann@oth-regensburg.de`
Datum:	21.03.2024
Abgabe bis:	27.03.2024 24:00

In-class exercises

The linux/unix command line[1] offers numerous versatile programs for processing, searching and systematically manipulating textual data (including big data). In this exercise, we'll use and learn about some of them.

An overview of GNU text processing programs can be found in `https://en.wikipedia.org/wiki/GNU_Core_Utilities`, a tutorial on using the command line can be found at `https://www.debian.org/doc/manuals/debian-reference/ch01.html` (specifically, you may want to take a look at Section 1.6).

Most command line programs are specialists that excel at one specific task. For complex operations, you'll often use multiple programs that feed into each other. Intermediate results are passed with the pipe operator (|).

Consult the help pages for the programs listed below. For example, you can type `man grep`.

C.1 Analysing lists of words

Please work on the following tasks in small groups of 3-4 students for about 15 minutes.

Pay attention (each one of you!) that you yourself understand what you as a group are doing and what your fellow students are discussion. (Otherwise you risk not learning as much as you could.)

We will discuss the results and resolve your questions afterwards. Do note down your questions while you are developing your results.

Please download `data.zip` from the Moodle course page.

[1]On Windows 10 and 11 you can use "WSL" (Sindows Subsystem for Linux). On Macos (and other BSD-like Unices) you will find programs that may differ in some specifics; if in doubt you can install the ones that are available under linux with `brew install coreutils`.

1. Use `wc` to count the number of male and female first names in the given lists. Which command line options can you use to print the number (and only this number)?
2. Use `grep` to check if your first name is contained in one of the lists.
3. Unsorted lists can be sorted with `sort`. Generate a sorted list of (female and male) first names. Are the original lists sorted?
4. Some first names can be used for both girls and boys. Try to use `uniq` (in combination with other tools) to find the number of names that apply to more than one gender.
5. the 'streaming editor' `sed` can be used for searching and replacing using regular expressions. Devise a strategy (including other tools) to deconstruct compound names (e.g. "Hans-Peter") into their constituents. How many 'simple' names are there in total?

C.2 Analysing texts into words

Please work on the following tasks in small groups of 3-4 students for about 10 minutes.

1. Choose any (longish) paragraph from any (longish) Wikipedia article (in German or English; make sure your group agrees on the paragraph).
2. Copy&paste it into a text file.
3. Discuss in the group what are words, what is whitespace and what punctuation.
4. Are there words that contain punctuation or words that contain whitespace? Are there boundaries between words without whitespace?

H 1 Take-home exercises

Please hand in your exercise solutions via the Moodle course.

For this first exercise sheet, each person submits individually.

H 1.1 Analysing lists of words

1. What is the command to find out if 'Jacky' is both a boys and a girls name according to our lists?
2. How many names in the lists are not compound names? (Use `grep` and a command line option.)
3. Is there a name that contains all vowel characters (a, e, i, o, u)? What is the corresponding regular expression? Discuss your result.

H 1.2 Segmenting texts into word tokens

Here, you'll use command line programms to segment texts into the words that it contains. Note for each step the command (or commands) that you use and report the results.

1. Use `sed` to transform punctuation in the text into blanks (i.e., what appears when you press the space bar on the keyboard).
2. Again, use `sed` to break lines at each sequence of one or more blanks. (Use \n as the character to be transformed to.)
3. How many words in total (tokens) and how many different words (types) does your text contain? How can you avoid counting words as two types when they appear in lowercase within a sentence but in uppercase at the sentence start?
4. Note down the ratio of tokens and types for one paragraph, three paragraphs and all text in your chosen Wikipedia article. Is there some tendency in the ratios?

H 1.3 Python and NLTK

1. Install a python3 environment as well as an IDE (I prefer PyCharm) on your favorite computing workplace (at the university or your own computer). Also, install the NLTK toolkit (from `https://www.nltk.org/` or using your paket manager, or using `pip`).
2. Tokenize the text in moodle (and/or your chosen Wikipedia paragraph) using the tutorial at `https://goodboychan.github.io/python/datacamp/natural_language_processing/2020/07/15/01-Regular-expressions-and-word-tokenization.html`.
3. Does the built-in word tokenization work as expected for German?

(For this exercise, it's sufficient to document the results of your experiments. I do not expect your results to be without error but want to see that you made an effort. It's particularly important that you exercise your python skills a bit.)