DATA.STAT.840 Statistical Methods for Text Data Analysis
Exercises for Lecture 3

Exercises for the remaining part of basic text processing

Exercise 2.4: Vocabulary pruning for Project Gutenberg books.
This exercise continues from exercise 2.2. In this exercise we will process top-downloaded public domain ebooks from Project Gutenberg. We assume steps (a)-(d) of exercise 2.2 have already been performed.

Prune the vocabulary to remove stopwords, overly short and long words, the top 1% most frequent words and words occurring less than 4 times. Report the top-100 words after pruning.

Exercises for collocations and vector spaces

Exercise 3.1: Lexical dispersion in Pride and Prejudice.

(a) Download the Project Gutenberg .TXT ebook of Jane Austen's "Pride and Prejudice", and process it using the pipeline as in Exercise 2.2 (a)-(d), no need to prune the vocabulary. (See https://en.wikipedia.org/wiki/Pride_and_Prejudice for information about the work.)
(b) Compute a lexical dispersion plot (or plots) for the following words: 'pride', 'prejudice', 'elizabeth', 'darcy', 'charlotte', 'love', 'hate', 'marriage', 'husband', 'wife', 'father', 'mother', 'daughter', 'family', 'dance', 'happy'. You can do this either as a single plot or multiple separate plots for different words or groups of words. Report the resulting plots.
(c) Discuss the results. Do the plots suggest something about the focus of the text in different parts of the book? You may also inspect the lexical dispersion of other words if you wish.

Report the resulting plots, your discussion, and the program code you used. You can do this exercise using Python or any other language, and using NLTK in Python or any other library.

Exercise 3.2: The concordances of Frankenstein.

(a) Download the Project Gutenberg .TXT ebook of Mary Wollstonecraft Shelley's "Frankenstein; Or, The Modern Prometheus", and process it using the pipeline as in Exercise 2.2 (a)-(d), no need to prune the vocabulary. (See https://en.wikipedia.org/wiki/Frankenstein for information about the work.)
(b) Create a concordance view of the following words each: 'science', 'horror', 'monster', 'fear'. Comment on the results.

Report the resulting plots, your code, and your comments.

(exercises continue on the next page)

Exercise 3.3: Collocations in Call of the Wild.

(a) Download the Project Gutenberg .TXT ebook of Jack London's "The Call of the Wild", and process it using the pipeline as in Exercise 2.2 (a)-(d) and the vocabulary pruning in Exercise 2.3. (See https://en.wikipedia.org/wiki/The_Call_of_the_Wild for information about the work.)

(b) Use the approach presented on Lecture 3 to count distance statistics between all pairs of words in your pruned vocabulary. You can use code from the lectures, or any other implementation in a library.

(c) Report the overall 20 collocations with smallest mean absolute distance, that are either "noun noun" or "adjective noun" type, and for each report the p-value of a significance test whether the mean absolute distance of the collocation differs from the mean absolute distance of a random word pair. Discuss the result: what does it tell you about the story content?

(d) For the word 'dog', report the 20 collocations with smallest mean absolute distance, that are either "noun noun" or "adjective noun" type, and for each report the p-value of a significance test whether the mean absolute distance of the collocation differs from the mean absolute distance of a random word pair. Discuss the result: what does it tell you about the story content?

Report the resulting statistics, your code, and your comments. You can do this exercise using Python or any other language, using NLTK in Python or any other library, and using the code on the lecture or any other available functions.

Exercise 3.4: Regular expressions of Frankenstein.

Use the Python regular expression syntax to find occurrences of the phrase "for ... years" where ... denotes one or more words in the middle of the phrase in "Frankenstein; Or, The Modern Prometheus". Print the resulting matches.

- For this exercise, you do not need to (and should not) do most of the preprocessing of Exercises 2.2 and 2.3, since regular expression search operates on the original string of letters of the text.
- This particular use case was not discussed on the lecture: you will need to read more about Python regular expressions here: https://docs.python.org/3/howto/regex.html . Hint: since you want to allow words of arbitrary length between "for" and "years", consider repeating patterns.

Report the matches you found and your program code. You also do this exercise in another language or using another regular expression library.

Exercise 3.5: Other document features.

For each "other document feature" category suggested on slides 39-41 of Lecture 3, suggest an additional feature.