KED2022 Assignment 3: Python NLP

Alex Flückiger | University of Lucerne

21 Oktober 2022

Requirements

- Deadline: 20 May 2022, 23:59
- File format: Jupyter Notebook (`.ipynb`) or executable Python script (`.py`)
- Naming schema: `SURNAME_KED2022_3.ipynb` or `SURNAME_KED2022_3.py` Replace `SURNAME` with your surname.
- Tasks require Python commands only, not Bash.
- Submit your solutions on time via the respective exercise module on OLAT. The module is only open until midnight.
- Find solutions individually. When you are stuck, post your issue in the OLAT forum and ask friends. In terms of programming, Google may be your best friend.

Introduction

You conduct your first substantial NLP analysis by comparing how the language differs between two social groups and periods.

In practice, you usually work with a collection of texts rather than a single document. A corpus may comprise from a handful up to millions of documents. In this exercise, you analyze an unpublished dataset[1] of historical 1 August speeches by Swiss Federal Councilors. Although this dataset is provided, you could create your custom dataset and analyze it the same way.

There are two options in this assignments. You either compare how the language changed over time or between Federal Councilors of different gender.

1 Corpus of 1 August Speeches by Swiss Federal Councilors

The aim is to compare the vocabulary across two groups of documents. You may compare speeches given before the turn of the millennium and those given after. Alternatively, you may compare the language between female and male Federal Councilors (see next page).

1. Make sure that your local copy of the GitHub repository `KED2022` is up-to-date with `git pull`. You find the dataset as `.csv` file here: `materials/data/dataset_speeches_federal_council_2019.csv`. Have a look at the file in a spreadsheet editor (e.g., Excel, Libreoffice Calc).
2. Open VS Code and create a new notebook (`.ipynb`).
3. Import the necessary modules in your script. You may subsequently add the modules as soon as you use them at the top of your script. Don't add modules that you never use.

[1] The historic dataset contains speeches given by Swiss Federal Councilors on the Swiss National Day. Simon Schmid (journalist at Republik), with the collaboration of Prof. Andreas Kley (Faculty of Law, UZH), collected many of these speeches and kindly shared the resulting dataset with me. The collection comprises 166 speeches; a multiple of the publicly available here.

4. Create a corpus object of the provided dataset using the function provided on the next page. You should copy it from `materials/code/KED2022_10.ipynb` rather than from this PDF due to formatting issues.
5. Create two subcorpora from the main corpus by filtering according to the following criteria (meta attributes `Sprache`, `Jahr`)[2]
    1. all German speeches hold before 2000
    2. all German speeches hold as from the year 2000
6. Print out the number of documents for both of the subcorpora.
7. Export the lowercased unigram vocabulary for each subcorpus to an individual file.

    Bonus: Set an additional argument in the function to get the relative frequency: `weighting="freq"`
8. Compare both vocabularies. How did the language change between these periods? Which terms have been the most popular in which period? Keep in mind that the absolute frequency is also dependent on the number of documents, not just the popularity of a term.
9. Summarize your impression in a few lines.

[2]To combine multiple criteria, simply use the logical `and` operator: `filter_func = lambda doc: doc._.meta.get("year") > 1900 and doc._.meta.get("year") < 2000`

1.1 Alternative: Gender instead of History

Are you rather tired of history and more interested in gender? Alternatively, you can divide the corpus by gender. Nevertheless, you should limit to German speeches hold from 1999 onwards for historical reasons[3] (meta attributes `Geschlecht`, `Sprache`, `Jahr`).

1.2 Function to read Dataset

```
def get_texts_from_csv(f_csv, text_column):
    """
    Read dataset from a csv file and sequentially stream the rows,
    including metadata.
    """

    # read dataframe
    df = pd.read_csv(f_csv)

    # keep only documents that have text
    filtered_df = df[df[text_column].notnull()]

    # iterate over rows in dataframe
    for idx, row in filtered_df.iterrows():

        # read text and join lines (hard line-breaks)
        text = row[text_column].replace('\n', ' ')

        # use all columns as metadata, except the column with the actual text
        metadata = row.to_dict()
        del metadata[text_column]

        yield (text, metadata)

f_csv = '../materials/dataset_speeches_federal_council_2019.csv'
texts = get_texts_from_csv(f_csv, text_column='Text')

corpus_speeches = textacy.Corpus(de, data=texts)
```

[3] In 1999, Ruth Dreifuss was the first woman who gave a speech in the role of a Swiss Federal President. Until 2004, only speeches by Swiss Federal Presidents are included in the dataset. After that year, speeches by Swiss Federal Councilors are covered as well. Thus, it would not be sensible to consider documents from the distant past as there are no female records.

2 Test your script

This task is just a sanity check for your script. Restart the environment to remove intermediate objects (e.g., variable) and, subsequently, run your code in one go by following these steps:

1. Click `Restart` in the menu bar at the top of the code
2. Click `Run All` next to it

Visual Studio Editor executes all the cells with code, one after another. Everything should be reconstructed accordingly and you should see a green tick below each cell. If not, correct the script so that it runs without any issue.

3 Feedback

Please answer the following questions at the end of your script. Start your answers with the `#` symbol to make them comments that are ignored when running the script.

1. Do you have any questions concerning the exercise or the commands?
2. How long did it take to solve this exercise? Give a fair estimation.