Code Manual

This manual is intended to provide a quick outline of our code (and data) files, mainly to demonstrate in which order we have conducted our steps, such that these steps are easily reproducible. It is NOT intended to serve as an excessive content-related manual (content is discussed in the paper [and in the comments of the code]).

1. Web-Scraping

We scrape personal information from the Bundestag website in the *bundestag_website.ipynb* file and append Twitter accounts scraped from party homepages in the *twitter_accounts.ipynb* file, yielding the *abg_df.pickle* dataset. We gather socioeconomic and election data in the *socioeconomic.ipynb* file, storing the resulting dataset in *se_df.pickle*. Filtering *abg_df.pickle* for parliamentarians that have a Twitter account, we download tweets for each parliamentarian in the file *tweepy_download.ipynb* (using *tweepy*) and store the results in *tweepy_df.pickle*. Note that we have written the download function for *tweepy* in a separate script, *tweepy_helpers.py*, which is imported at the beginning of *tweepy_download.ipynb*. A quick demo covering twitter scraping can be found in *twitter_scraping_demo.ipynb* (we also demonstrate how *getoldtweets3* can be used for twitter scraping instead of *tweepy*). Finally, in *stm_prep.ipynb* we prepare all python files for our subsequent analysis in R by converting pickle files to csv files. Note that the file containing the tweets, *tweepy_df.pickle*, is already very large (almost 1 GB) and converting it to csv format increases its size to 5 GB. Therefore, only *tweepy_df.pickle* is available in the data folder, while *tweepy_df.csv* must be generated by the user on a local machine.

The code files, as handed in along with this manual, represent the data collection process as of April 2020. The websites scraped, in particular the Bundestag website and party homepages, are constantly updated according to changes in the composition of the German parliament. Therefore, our results are not exactly reproducible. For instance, the official FDP party homepage is now organized differently, so that the user would need to update the corresponding code chunks in *twitter_accounts.ipynb* (currently commented out for runnability of the remaining code). All other code files are runnable without further adjustments. Note that whenever a selenium web driver is used, timeout errors might occur; in that case, the user must rerun the missing parts.

2. Preparation

Code file: *topic_preparation.R*
We start by converting *tweepy_df.csv* to an rds-file, *topic.rds*, which is MUCH smaller. In doing so, we also remove all observations which we do not need for our specific analysis, such as retweets and tweets prior to September 24, 2017. If the user does not wish to create the 5 GB *tweepy_df.csv* file, we recommend to simply start with *topic.rds*.

We then proceed by merging personal level-data as well as structural covariates to the Twitter data (that is, merging *abg_df.csv* and *se_df.csv* with *topic.rds*) and aggregate our data on a monthly level. Furthermore, we create train-test splits of our data. We then save the different versions (i.e., non-monthly, monthly, training data, test data) in the folder *topic_preparation*, in order to proceed with preprocessing, which must be conducted separately for each data set.

3. Preprocessing

Code file: *topic_preprocessing.R*
In this section, we conduct preprocessing for the different data sets previously created. We describe this procedure in detail in our paper. Preprocessing comprises removing stopwords, converting German umlauts, etc. At the end of preprocessing we obtain a document-feature matrix (DFM) for each of our data sets. Note that preprocessing is computationally expensive and might take a while. We store the preprocessed data in the folder *topic_preprocessing.*

4. Section-specific files

Code for sections 4-6 is found in the R-files named according to the respective section. In these files, we conduct calculations and generate figures for the corresponding sections. Note that we have stored all important results in the respective data folders (also labelled according to each section); in our R-code we commented out the parts where these results were obtained, such that they are not overwritten by accident (and, in addition, due to long runtimes).