Homework III

PLSC 498 – Text as Data

*Professor Kevin Munger*

This homework is due electronically by **11:59 p.m. EST on November 19, 2022**. You can submit your homework on Canvas. Late work will incur penalties of the equivalent of one third of a letter grade per day late.

**Please use your own data (or a subset of it) for the final project to complete this assignment.**

It must be your own work, and your own work only—you must not copy anyone’s work or allow anyone to copy yours. This extends to writing code. You may consult with others, but when you write up, you must do so alone.

Your homework submission must be in one of the following formats: (1) A set of answers and a clearly commented R code appendix (use comments to identify code relevant to each answer you produced), (2) A report consisting of clearly marked answers, each accompanied by the relevant code (e.g., a report generated using *rmarkdown, knitr,* or similar). **In either case, your code must be included in full, such that your understanding of the problems can be assessed.**

PART 1

1. **Applying topic models** to your own data:
   1. To decrease the time it takes to fit a topic model, we will limit our analysis to a subset of your data. Create a subset of your data as a corpus.
   2. Create a document term matrix with your new corpus in which punctuation and numbers are removed and words are stemmed and set to lower case. Also, the standard English stopwords. Finally, use quanteda’s “dfm trim” to remove words that occur fewer than 30 times or in fewer than 10 documents. Report the remaining number of features and the total number of documents in the DFM.
   3. Preprocessing decisions can have substantive impacts on the topics created by topic model algorithms. Make a brief (1 paragraph) argument for or against removing rare terms from a dfm on which you plan to fit a topic model.
   4. Fit a topic model with 30 topics using LDA(), with method = ”Gibbs”. Increase the number of iterations to 3000 to ensure that the model describes

the underlying data well and set the seed to 10012 so that you can replicate your results. Report the loglikelihood of your topic model object.

* 1. Examine the top 10 words that contribute the most to each topic using get terms(). Find the most likely topic for each document using topics(). Rank topics according to the number of documents for which they are the most likely topic and label the top five (i.e. by looking at the most likely words within each of these topics). Explain your choice of labels. You should save the top 10 words over all 30 topics, for later use.

1. **Topic stability:** We want to see how stable these topics are, under two different topic parameter values.
   1. Re-run the model from question 1 with a different seed. Report the @log- likelihood of your topic model object.
   2. For each topic in the new model, and the topic that is the closest match in the original run in terms of the highest number of the same terms in the top 10.
   3. Calculate the average number of words in the top 10 words shared by each matched topic pair. Your answer should be a table.
   4. Now run two more models, but this time, use only 5 topics. Again, find the average number of words in the top ten shared by each matched topic pair. How stable are the models with 5 topics compared to the models with 30 topics?
2. **Topic Models with covariates:** The Structural Topic Model (STM) is designed to incorporate document-level variables into a standard topic model. We can use an STM (from the stm package) to model the effects of these covariates directly.
   1. Describe one covariate that you think is important about your data. Make sure that it is a binary variable, or else transform it so that it is binary (ie set all values above the median to 1, below the median to 0).
   2. Fit an STM model where the topic content varies according to this binary variable. Be sure to use the spectral initialization and set k=0, which will allow the STM function to automatically select a number of topics using the spectral learning method.

Keep in mind that this function is computationally demanding, so start with the minimum threshold document frequency threshold set to 20; if your computer takes an unreasonably long time to fit the STM model with this threshold, you can raise it to as high as 30. Report the number of topics selected in the fitted model. Also report the number of iterations completed before the model converged.

* 1. Identify and name each of the 5 topics that occur in the highest proportion of documents using the following code: plot(fit.stm, type = ”summary”)