Homework 3

DS-GA 3001-004, Text as Data

Prof Arthur Spirling

This homework must be returned to Kevin Munger's box on the 2nd floor of 19 West 4th Street by **noon on Monday, May 9th, 2016**. Late work will incur penalties of the equivalent of one third of a letter grade per day late.

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

You must include an `R` code appendix which is clearly commented, such that it your understanding of the problems can be assessed. You must turn in a paper copy: **no electronic copies will be accepted**.

(Exercise 1 and based on one prepared by Kenneth Benoit and Paul Nulty). In this exercise we will use the R packages `topicmodels`, `lda` and `stm` to discover topics in a set of UK newspaper articles relating to immigration.

0. **Setup** We will begin by loading the packages and data necessary and converting our corpus into the necessary format.

    a. The corpus of newspaper articles is part of the `quantedaData` package, and is named `immigNewsCorpus`. Load and examine this corpus.

    ```
    require(quanteda, warn.conflicts = FALSE, quietly = TRUE)
    data(immigNewsCorpus, package = "quantedaData")
    ```

    b. Make sure that the `lda`, `topicmodels`, `stm` and `LDAvis` packages are installed.

1. **topicmodels** and the news corpus.

    a. Building the topic model on large corpora with many topics can take a long time. We will select only the four most common newspapers from the corpus.

    ```
    summary(immigNewsCorpus, 5)
    ```

    ```
    ## Corpus consisting of 2833 documents, showing 5 documents.
    ```

    ```
    ## Warning in nsentence.character(object, ...): nsentence() does not correctly
    ## count sentences in all lower-cased text
    ```

    ```
    ##   Text Types Tokens Sentences paperName day   id
    ##  text1   524   1067        31   express 100 3198
    ##  text2   261    451        30   express 104 2425
    ##  text3   367    651        33   express 104 2427
    ##  text4   131    194        19   express 104 2428
    ##  text5   306    511        29   express 106 2376
    ##
    ## Source:  /home/paul/Dropbox/code/quantedaData/* on x86_64 by paul
    ## Created: Tue Sep 16 16:36:41 2014
    ## Notes:
    ```

```
topPapers <- sort(table(immigNewsCorpus[["paperName"]]), decreasing = TRUE)
reducedCorpus <- subset(immigNewsCorpus, paperName %in% names(topPapers)[1:4])
table(reducedCorpus[["paperName"]])
```

```
##
##  guardian      mail telegraph     times
##       392       412       511       352
```

b. Create a dfm, but don't remove the basic stopword list. Instead, load the R file called `custom_stopwords` from the homework folder and remove those. To make sure you've got the right list, run this line of code to see the words we're talking about, such as copyright notices and references to the paper and other articles.

```
custom_stopwords[573:588]  # examples of the custom stopwords
```

c. Now remove words that occur fewer than 30 times or in fewer than 20 documents; use `quanteda`'s "trim" function.

d. Fit a topic model with 30 topics using `LDA()`, with `method = "Gibbs"`. Be sure to include enough iterations of the algorithm. This might take a few minutes. Be sure to set.seed.

e. Examine the words that contribute the most to each topic (examine the top 10). Use `get_terms()`. Choose a name for the top 5 topics, and explain your choice. Be sure to save the top 10 words from all 30 topics, for later use.

f. Examine the topics that contribute the most to each document, using the code from Recitation_8 to visualize the top two topics per document. Instead of dividing the documents into coalitions, just look at the top two topics for all of the documents published by the Guardian and the Daily Mail. Discuss your findings.

```
doc_topics<-TM[["Gibbs"]]@gamma
```

g. Finally, we can find the average contribution of a topic to an article from a particular newspaper, and compare newspapers on particular topics. For each of the topics you've named, see how their prevalence varies among the different newspapers. Discuss your findings.

h. Use LDAVis to present the results of the topic model fit. You can just take a screenshot of the results of LDAVis. Discuss your results; what have we learned about the newspapers?

2. **Topic Stability**: We want to see how stable these topics are, under two different sets of pre-processings.

   a. Re-run the model from question 1, but choose a different seed.

   b. For each topic in the new model, find the topic that is the closest match in the original run in terms of cosine similarity of the topic distribution over words.

   c. Calculate the average number of words in the top ten SHARED by each matched topic pair.

   d. Now run two more models, but this time, use only ten topics. Again, find the average number of words in the top ten SHARED by each matched topic pair. How stable are the models with ten topics compared to the models with 30 topics?

3. **Topic Models with covariates**: We have information about both the newspaper and the date of the articles, so we can use the Structural Topic Model to model the effects of these covariates directly.

   a. Construct a useful date variable from the variable `reducedCorpus$documents$day`.

   b. Let's only look at the articles written by the Guardian and the Daily Mail. Fit an STM model where the topic content varies according to this binary variable, and where the prevalence varies according to both this binary variable and the spline of the date variable you've created. Be sure to use the spectral initialization and set k=0. If your computer can't handle this, you can raise the minimum threshold document frequency for the terms.

c. Look at the top 5 topics and name them.

d. Using the visualization commands in the `stm` package, discuss one of these top 5 topics. How does the content vary with the paper discussing that topic? How does the prevalence change over time?

4. **Non-Parametric Scaling: Wordfish**: We'd like to be able to scale political texts according to some latent dimension, and Wordfish allows us to do so. Becuase the newspaper corpus does not have an obvious latent dimension, we'll return to that old standby: the State of the Union addresses.

   a. Run the following code to get all SOTUs after 1900:

```
r mydfm <- dfm(subset(inaugCorpus, Year>1900))
```

   b. Wordfish requires us to anchor two documents at the extremes: choose Obama's 2009 speech as the most left-wing, and Nixon's 1969 as the most right-wing.

   c. Which of the scaled documents is the most right-wing? The most left-wing?

   d. Look at the estimated Beta value for the word `fascism`. Does this word place a document on the left or on the right?

   e. Re-create the "guitar plot" from Recitation 7. Explain what this plot is designed to show.