PLSC 497 Text as Data Prof. Kevin Munger

Assignment date: January 29, 2021

# Practice Homework

This homework is due electronically by **11:59 p.m. EST on Wednesday, February 10, 2021**. You can submit your homework by **emailing copies** **both** to Prof. Munger (kmm7999@psu.edu) and Mr. Villegas-Cruz (amv5718@psu.edu). 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 al- low 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.

# Conceptual Questions:

Question 1) What are latent variables?

Latent variables are variables that cannot be directly observed but can be inferred through the model from observable variables and could be related in Text Analysis to author characteristics, document features, or both.

Question 2) What is stemming? How is it different from lemmatization?

To reduce the complexity of the unstructured text, during the process of stemming, multiple forms of the same word are converted into a single stem. In contrast to it, instead of simple reduction of the word, lemmatization considers full vocabulary and applies the morphological analysis of the word, creating a lemma. While simple and fast stemming can be efficient, the lemmatization is more informative and looks at the surrounding text to determine the word‘s part of speech.

Question 3) What is a document term matrix? Why is it usually sparse?

A document-Term matrix is a mathematical matrix that describes the frequency of the term that occurs in the collection of the documents. Within the matrix, each word is a column, and which document is one document. The values are the number of times a given term appears in a particular document. The sparse format is more efficient as many text mining tools and algorithm works with only sparse document-term matrices. Also, they are constructed from weighted term-frequency vectors.

Question 4) Explain the tf-idf statistic and the advantage of using it:

Vectorized Term Frequency and Inverse Document Frequency strategy of text mining converts the document into the bag of words and assign weighted terms to each word, calculating the weighted frequency of the term of a word in a document. It is simple and easy to compute. At the same time, it creates basic matrices to extract the most descriptive terms in a document. It also allows computing similarity between two documents.

# Coding Tasks:

Question 1) Use the Quanteda R package and load in the corpus of presidential inaugural addresses, 'data\_corpus\_inaugural'. Summarize the corpus.

```{r, include = F}

install.packages("quanteda") #instal package quanteda

library(quanteda) #load the package quanteda to access the corpus of presidential addresses

library(dplyr) #load the package dplyr to use glimpse function later

```

```{r}

data("data\_corpus\_inaugural") # load in the corpus needed

glimpse(data\_corpus\_inaugural) “#show the structure of the corpus

summary(data\_corpus\_inaugural) #summary of the corpus

```

Question 2) Using the docvars function, save the last name of the presidents in a vector.

```{r}

x<-tail(docvars(data\_corpus\_inaugural, "President"), 58) #create a vector which contains 58 presidents’ last names from the corpus

x #show the created vector

```

Question 3) Create a document term matrix (aka document feature matrix) to create a matrix of counts of the occurrences of each word in each document. Report the dimensions of this matrix.

```{r}

#install.packages("tidytext") install tidytext to use tidy function

library(tidytext) # load tidytext

presidential <- quanteda::dfm(data\_corpus\_inaugural) %>% # creates a document feature matrix “inaugural”

tidy() #turns document-term matrix into one token per row matrix

presidential #shows the matrix

```

Dimensions: 44710 obs. of 3 variables