DS5110 HW 6 - Due April 3

Kylie Ariel Bemis

3/20/2020

Instructions

Create a directory with the following structure:

- `hw6-your-name/hw6-your-name.Rmd`
- `hw6-your-name/hw6-your-name.pdf`

where `hw6-your-name.Rmd` is an R Markdown file that compiles to create `hw6-your-name.pdf`.

Do not include data in the directory. Compress the directory as `.zip`.

Your solution should include all of the code necessary to answer the problems. All of your code should run (assuming the data is available). All plots should be generated using `ggplot2`. Missing values and overplotting should be handled appropriately. Axes should be labeled clearly and accurately.

To submit your solution, create a new private post of type "Note" on Piazza, select "Individual Student(s) / Instructor(s)" and type "Instructors", select the folder "***hw6***", go to Insert->Insert file in the Rich Text Editor, upload your `.zip` homework solution. Title your note "[hw6 solutions] - your name" and post the private note to Piazza. **Be sure to post it only to instructors**

Part A

Problems 1–3 use the text of 56 major speeches by Donald Trump from June 2015 through November 2016. Download the data from https://github.com/PedramNavid/trump_speeches. (Use either the "full_speech.txt" file or the "speech_##.txt" files but not both, as the former contains all of the text of the latter. If you use the "speech_##.txt" files, you should skip the first line of each file.) Use the `read_lines()` function from the `readr` package to import the data into R.

Problem 1

Import the text from all 56 Donald Trump speeches into R. Tokenize the data into a tidy text data frame, using *bigrams* as tokens. Filter the data, removing bigrams where either word is a stop word or the word "applause", and removing bigrams where the first word is a negation word such as "never", "no", "not", or "without". Then plot the top 15 most common bigrams in Trump's speeches.

Problem 2

We would like to see the most commonly negated words in Donald Trump's speeches, and how they're negated. Filter the bigrams, keeping only bigrams where the first word is any of "not", "no", "never", or "without", and removing those where the second word is a stop word or "applause". Then visualize the most common (top ~5) words preceded (separately) by each of "never", "no", "not", and "without".

Problem 3

We would like to do a sentiment analysis of Donald Trump's speeches. In order to make sure sentiments are assigned to appropriate contexts, first tokenize the speeches into bigrams, and filter out all bigrams where the first word is any of the words "not", "no", "never", or "without".

Now consider only the second word of each bigram. Filter out all bigrams where the second word is a stop word or "applause". Then visualize the most common words (top ~5) in Trump's speeches that are associated with each of the 6 sentiments in the "loughran" lexicon.

Part B

Problems 3–5 use data from Project Guternberg. Install the `gutenbergr` package and use the `gutenberg_works()` function to access the data.

Problem 4

Use the `gutenberg_works()` function to download the novels *Pride and Prejudice* by Jane Austen and *War of the Worlds* by H. G. Wells.

Visualize the top 15 most common words in each novel after remove stop words and words that appear less than 10 times total in the dataset.

Which words are indicative of Jane Austen's novel? Which words are indicative of H. G. Wells?

Problem 5

Treating each line of text (each row) as a separate document, transform the dataset into a document-term matrix, and then split it into a training and test set.

Use the `caret` package to fit a classification model of your choice to classify the author of each line of text. Report your accuracy on your test set.

Hints: After transforming to a document-term matrix, you can use access the `rownames()` of the output matrix to get the document IDs. For most methods, you may also need to coerce the sparse matrix into a dense matrix with `as.matrix()`. You will probably want to use the train() method that takes separate `x` and `y` parameters rather than use the formula interface.