This paper provides the steps to create a wordcloud with R-studio using a work that is part of the Project Gutenberg collection which makes classic texts available for free. In addition, it includes the steps involved in creating a latex document (what you are reading).

1one

The set up.

In order to create our word cloud in R-Studio we first have to install the various packages we’ll use to extract the data, create our word cloud and then to format everything into our latex document.

We’ll need to install the following packages: dplyr, , tidytext, stringr, wordcloud, tm, gutenbergr, and knitr. If you already have the packages installed, you can skip to the next step and library each package into R-Studio. Once you have all the packages installed and library-ed, open a new R script file.

2two

Finding our text

We will be using the gutenbergr package to retrieve a text. The gutenbergr package allows us to download and process public domain works in the Project Gutenberg collection <http://www.gutenberg.org/>. Includes metadata for all Project Gutenberg works, so that they can be searched and retrieved. Footnote(<https://cran.r-project.org/web/packages/gutenbergr/index.html>). We’ll be looking for Wuthering Heights by Emily Bronte.

The following code also uses stringr so that we don’t have to have the information exactly as it is the data. We will need to add the libraries for stringr & gutenbergr to our code so that it will run.

gutenberg\_works(str\_detect(author, "Brontë"))

We've got more than one text listed so we'll need to refine our search:

The number `768’ is the Gutenberg identification number or gutenberg\_id. We’ll use this number to download our text.

Now that we have the right book, we’ll save this into our data frame variable wh.

Now we'll take a quick look at our data by looking at the head \& tail:

It looks like the data loaded

Even though we don’t have to, let’s get rid of the words ‘chapter’. In order to do this, we’ll need to load the libraries for tidytext and tm (which stands for text mining).

We can see in the R-studio console that the word chapter (that we could see before in line 3 has been removed. We’ll save this into our wh data frame variable.

Again, let's take a quick look at our data

We can also see that lines 1-5 only contain the title & all the rest are blank lines. Let’s get rid of these as well.

We can also see that lines 1-5 only contain the title \& all the rest are blank lines. Let’s get rid of these as well. The wh[1:6,] is calling the row numbers 1 to 6. The space after the comma would be if we wanted to limit the rows.

We’ll need to know the total number of lines so let’s get the dimensions of our data frame:

So now that we know our dimensions, we’ll save only the line numbers we want to keep into our wh data frame variable.

Now when we call the wh variable, we see that all the blank lines are gone. Note that the lines are re-numbered as well. Again, we can see this in the R-studio console.

Now we want to break the lines of text into individual words. We use the unnest function from tidytext. We'll bring in the library for tidy text and tm (text mining) . The unnest needs (input, output). For our output (which we'll call word) it will make a column called `word'. The input (which we'll call text) will be the individual words.

We’ll save this back into our wh variable

We’ll save this back into a new variable called words\\_wh

Now that we have the individual words we just need a couple more steps before we can make our word cloud. We want to get rid of all the ‘stop words.’ These are the ‘small’ words like in, the, a, an, at , to, etc.

Again, we’ll save these all back into our variable words\\_wh

Getting Ready for Our Wordcloud

Now that we only the ‘main’ or important words, we’ll group these words together to get the frequency of each word.

Now we are ready for our wordcloud

Now we are ready for our wordcloud. We'll need to bring in the wordcloud library. The minimum frequency number helps to determine the size. If we use a larger number (like 50 or 100) it makes our word cloud smaller (less words). A smaller number gives us more words, but many of the words are smaller. Basically we need to ‘play’ with the min.freq number until we get something that looks appealing.

Looks pretty good, but let’s see if we can add some color. Again, you need to ‘play with the color numbers and color names to find an appealing cloud.

Feinerer, I. and Hornik, K. (2017). tm: Text Mining Package. R package version

0.7-1.

Fellows, I. (2014). wordcloud: Word Clouds. R package version 2.5.

Robinson, D. and Silge, J. (2017). tidytext: Text Mining using 'dplyr', 'ggplot2',

and Other Tidy Tools. R package version 0.1.4.

Silge, J. (2017). janeaustenr: Jane Austen's Complete Novels. R package version

0.1.5.

4

Silge, J. and Robinson, D. (2017). Text Mining with R: A Tidy Approach.

O'Reilly Media.

Wickham, H. (2017). stringr: Simple, Consistent Wrappers for Common String

Operations. R package version 1.2.0.

Wickham, H., Francois, R., Henry, L., and Mller, K. (2017). dplyr: A Grammar

of Data Manipulation. R package version 0.7.4.

Wickham, H. and Grolemund, G. (2017). R for Data Science: Import, Tidy,

Transform, and Model Data. O'Reilly Media.

5