Blog for 10-25 tidytext part 2

We’ll need to install & library the following packages:

tidytext

tm (don’t need to library) -tm stands for text mining

wordcloud

dplyr

plus

janeausten

The janeausten is all of her novels as text.

Open a New script R file

We can’t quire remember what the data fram is in the janeausten library

When we start to type in ‘austen” into the command prompt we get a pop up box that tells us it is austen\_books() we can press F1 for help & we get more information in the “help” tab to the right of the console.

So if we hit return after > austen\_books() we see a data fram with char & factors.

So lets save this data frame into a variable:

sns<-austen\_books()

lets look at the head & tail in the command line:

> head(sns)

# A tibble: 6 x 2

text book

<chr> <fctr>

1 SENSE AND SENSIBILITY Sense & Sensibility

2 Sense & Sensibility

3 by Jane Austen Sense & Sensibility

4 Sense & Sensibility

5 (1811) Sense & Sensibility

6 Sense & Sensibility

> tail(sns)

# A tibble: 6 x 2

text book

<chr> <fctr>

1 possible, more distinguished in its domestic virtues than in its Persuasion

2 national importance. Persuasion

3 Persuasion

4 Persuasion

5 Persuasion

6 Finis Persuasion

We only want to worry about the book Sense & Sensibility. So let’s filter on it:

sns%>%

filter(book =='Sense & Sensibility')

Let” save this to a variable:

sns$book<- sns%>%

filter(book =='Sense & Sensibility')

If we type head(sns) and tail(sns) in the command line we see that we now only have the 'Sense & Sensibility' book

sns$book<-as.character(sns$book)

The as character $book converts any “non-character” format so that now we have all text.

We want to get rid of the Chapter Headings (Chapter 1, Chapter 2, etc)

First let’s check for the lines that begin with the word CHAPTER:

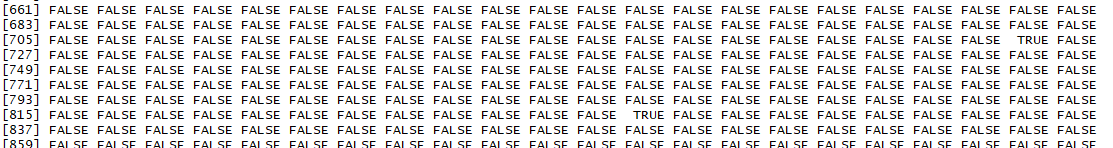
str\_detect(sns$text,'^CHAPTER')

the ^ up carat is begins with

Our output when we run the code is FALSE & TRUE (True for the Chaper headings).

A couple of these look close together (it doesn’t seem the chapters would be that short. So lets look at those individual lines

To the left is a number in brackets. It is the line number of the first item. So to find each of the lines, we need to count over to get the line number:



We are going to look at lines 725 & 827 as these “true” are pretty close together

#sns$text[725]

#sns$text[827]

When we run the code, we get:

> sns$text[725]

[1] "CHAPTER 5"

> sns$text[827]

[1] "CHAPTER 6"

>

So let’s see how many Chapter’s we have?

sns%>%

filter(str\_detect(sns$text,'^CHAPTER'))

we’ll save this to a temporary file & then look at the tail:

sns%>%

filter(str\_detect(sns$text,'^CHAPTER'))

> tail(temp)

# A tibble: 6 x 2

text book

<chr> <fctr>

1 CHAPTER 45 Sense & Sensibility

2 CHAPTER 46 Sense & Sensibility

3 CHAPTER 47 Sense & Sensibility

4 CHAPTER 48 Sense & Sensibility

5 CHAPTER 49 Sense & Sensibility

6 CHAPTER 50 Sense & Sensibility

Let’s google the number of Chapters… yep 50 chapters.

Now he code to remove those lines that start with CHAPTER:

Remember ! means not)

sns<-sns%>%

filter(!str\_detect(sns$text,'^CHAPTER'))

Let’s look at the first 20 lines:

Type into the command line:

> head(sns,50)

We see a bunch of stuff we don’t want before the actual text of the novel starts.

What we want starts on line 12.

We can look at the rows & columns of sns.

The code is variable [1:11, 2:4} the first set of numbers is rthe rows (rows 1 to 11) the second number is columns (columns 2 to 4).

We only have one column so we can leave the second per blank (but we need the comma)

Run the following:

sns[1:11,]

it shows the first 11 rows.

Lets see if there is anything we don’t want at the end.

If we use tail(0 it give the numbers 1-5 for the last 5 rows. We want to know HOW MANY rows we actually have. We need to know the dimension.

Run the following:

dim(sns)

it returns:

> dim(sns)

[1] 12574 2

>

There are 12,574 rows and two columns.

We know we don’t want the first 11 rows. So lets take those out of sns:

sns<-sns[12:12574,]

NOTICE we have the comma after the numbers.

Let’s look at the tail of sns

(type tail(sns)) into the command line:

> tail(sns)

# A tibble: 6 x 2

text book

<chr> <fctr>

1 within sight of each other, they could live without disagreement Sense & Sensibility

2 between themselves, or producing coolness between their husbands. Sense & Sensibility

3 Sense & Sensibility

4 Sense & Sensibility

5 Sense & Sensibility

6 THE END Sense & Sensibility

>

Ok it looks like we don’t need the last ‘THE END’ line.

Keep in mind that we’ve deleted the first 11 rows so our dimensions have changed.

> dim(sns)

[1] 12563 2

To get rid of ‘THE END’ change 12563 to 12562

Or delete 12 from the original 12574 to 12562

(don’t forget we took out 11 lines so we are back at line 1…)

Save this all into our sns variable.

sns<-sns[1:12562,]

Now this R script file is pretty “messy” with all our set up information. Let’s save this file as “set-up” and start a fresh file.

Now we’ll be using Tidy text to make our word cloud:

We will use unnest: Unnest makes individual words from our data frame. The unnest needs (input, output). For our output (which we’ll call words) this will make a column called ‘word’. The input (which we’ll call text) will be the individual words. The following shows us the words & the count of the number of times used.

sns%>%

unnest\_tokens(word,text)

We’ll save this into a variable words\_df:

words\_df<-sns%>%

unnest\_tokens(word,text)

We see words like a, an, the etc. These are called stop words in tidytext.

Let’s remove these (using that ! for not):

words\_df%>%

filter(!(word %in% stop\_words$word))

Let’s save this to our df variable:

words\_df<-words\_df%>%

filter(!(word %in% stop\_words$word))

We want to count the frequency of the words next. Group by word & use the count=n():

words\_df%>%

filter(!(word %in% stop\_words$word))%>%

group\_by(word)%>%

summarise(count=n())

We still have almost 6,000 rows:

1 1 1

2 200 1

3 7000l 1

4 abandoned 1

5 abatement 1

6 abbeyland 1

7 abhor 1

8 abhorred 2

9 abhorrence 4

10 abilities 9

# ... with 5,833 more rows

Let’s save this into our variable:

words\_df<-words\_df%>%

filter(!(word %in% stop\_words$word))%>%

group\_by(word)%>%

summarise(count=n())

NOW we’re going to save as the variable word\_freq:

word\_freq<-words\_df<-words\_df%>%

filter(!(word %in% stop\_words$word))%>%

group\_by(word)%>%

summarise(count=n())

When we try

wordcloud(word\_freq$word,word\_freq$count)

it takes too long -there are too many words.

If we type ?wordcloud into the command line we get some info (help)

Lets us the min.freq so words with frequency below min.freq will not be plotted.

Try:

wordcloud(word\_freq$word,word\_freq$count,min.freq=100)

this gives us a wordcloud but it took a long time & we get warnings that there are too many words (in the command line). It is also pretty congested, lets see if we can use a lower number for a better cloud.

Lets keep trying smaller numbers until we get a cloud we like:

wordcloud(word\_freq$word,word\_freq$count,min.freq=25)

gives us an OK cloud lower numbers are too few words.

Let’s push this all up to git hub so we don’t need notes.

fin