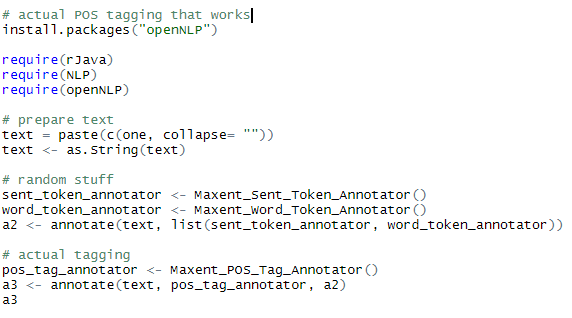
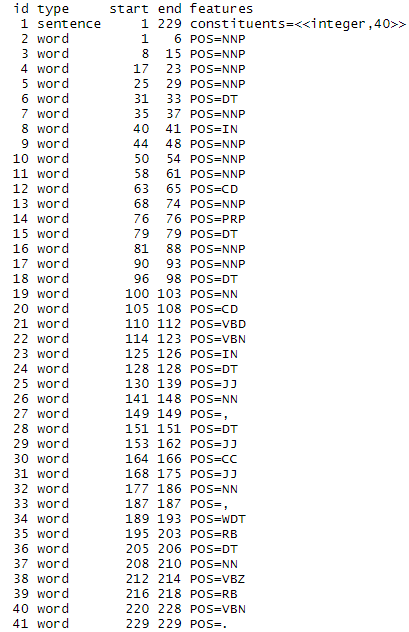
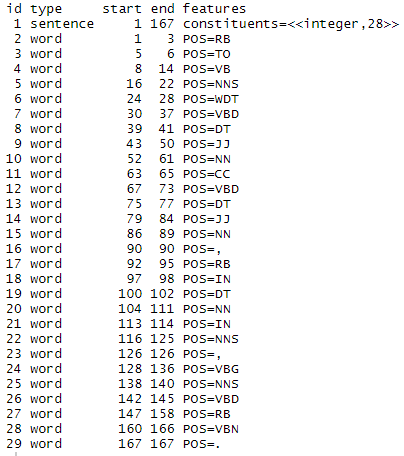
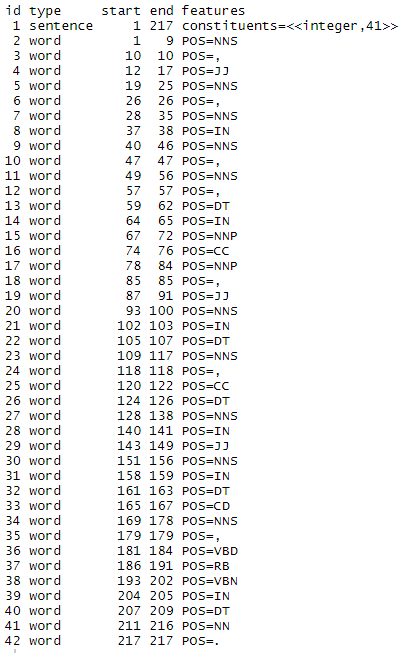
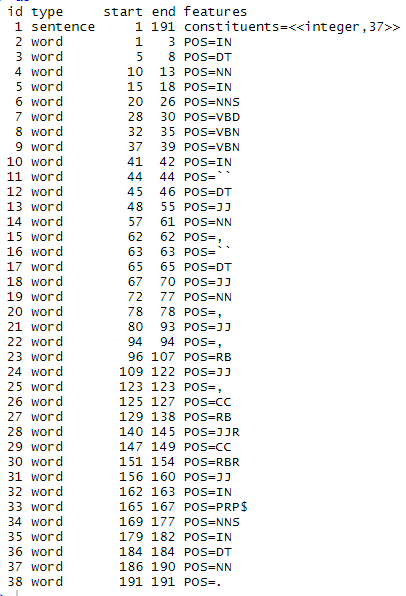
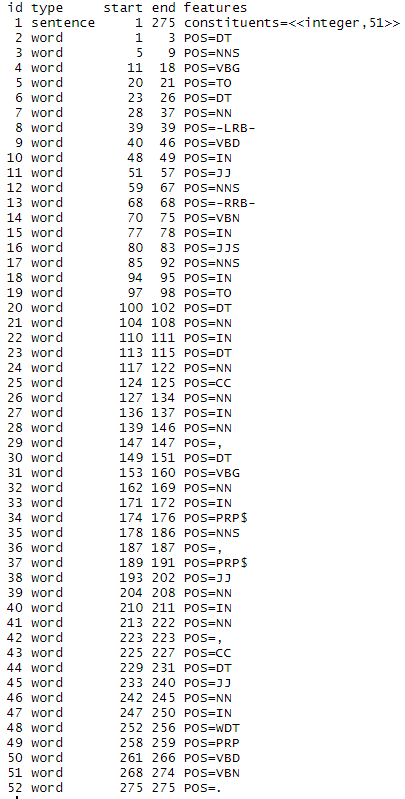
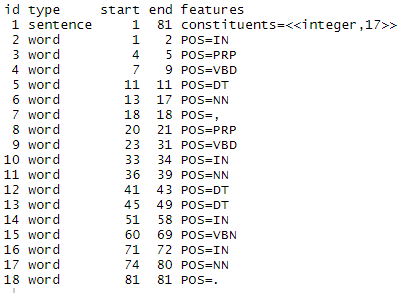
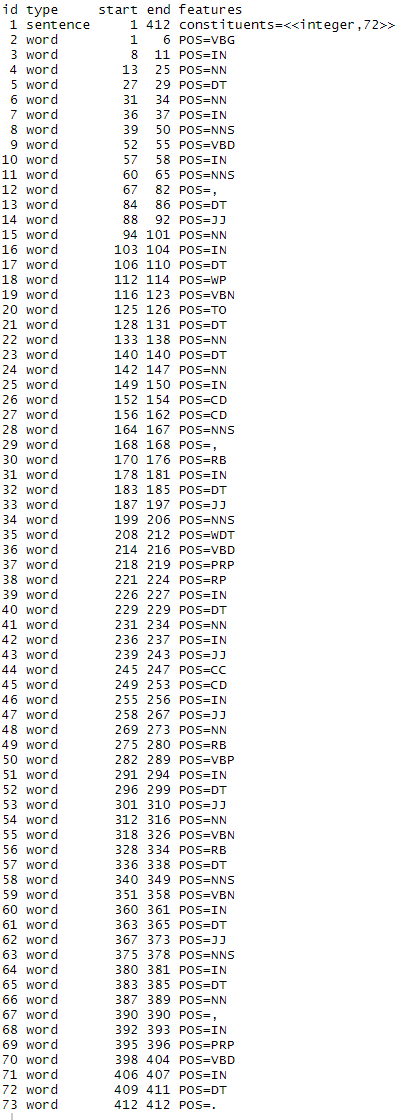
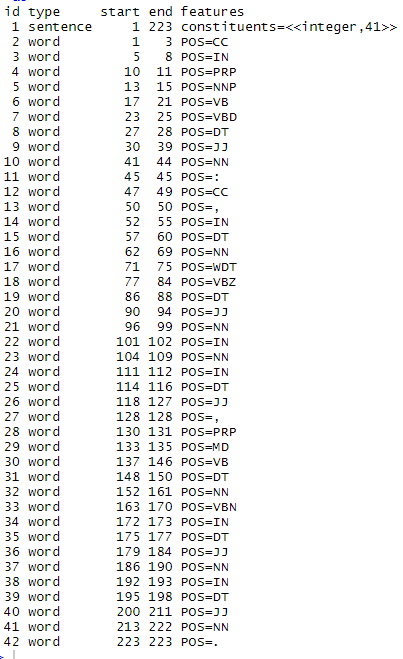
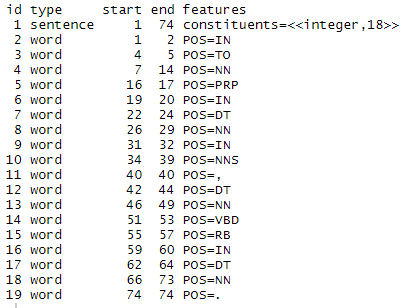
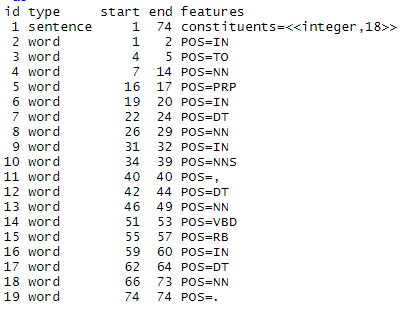
d.

For part D of the project, I found it best to use the openNLP library rather than wordnet. Wordnet works in terms of analyzing synsets, but it doesn’t function as well as openNLP for analyzing parts of speech within the context of the sentence. This makes NLP challenging because one word could have different POS depending on the context.

Code:

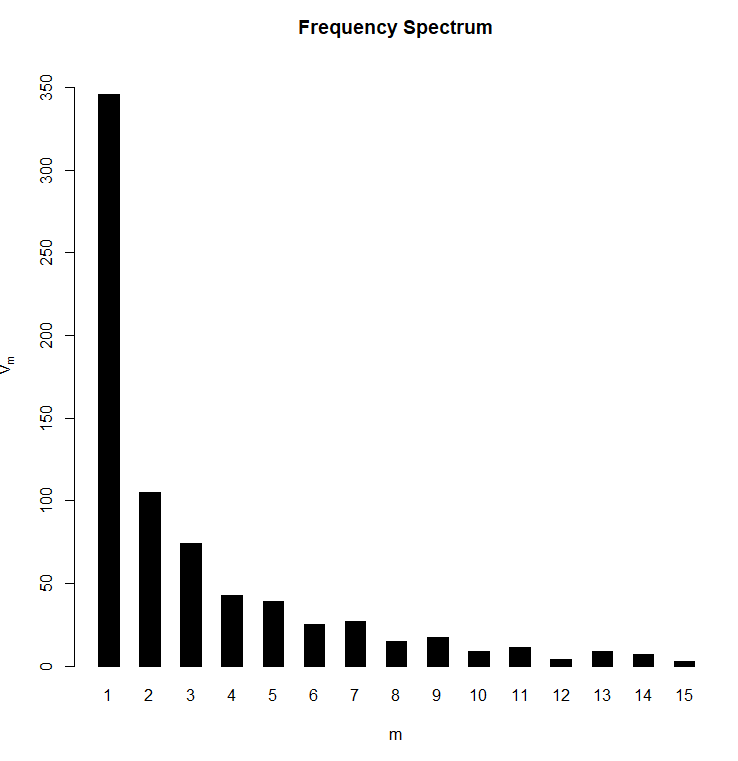


For the following examples, the ten longest sentences found above were run through the POS taggers using the code directly above. The sentences are ordered as they are above, with the POS in the last column of these print blocks

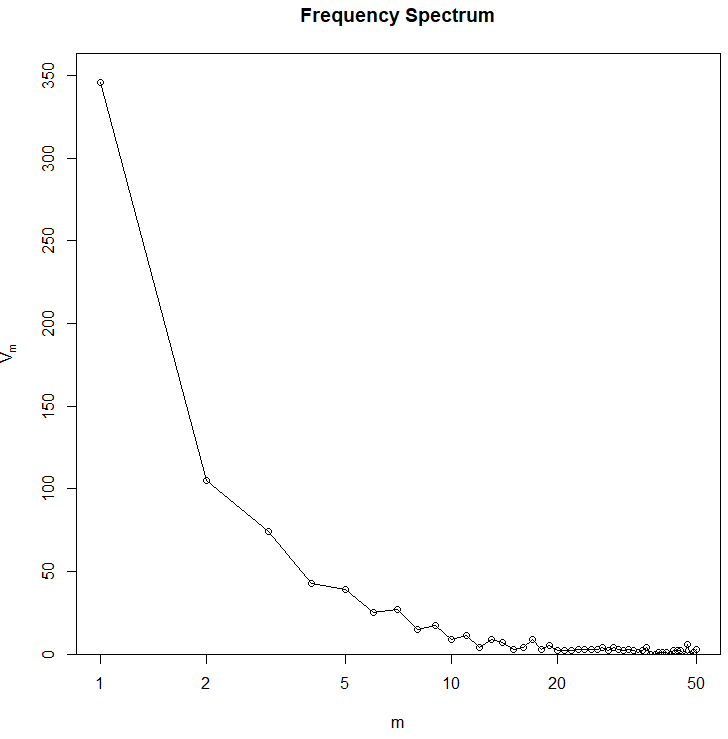
1. 
2. 
3. 
4. 
5. 
6. 
7. 
8. 
9. 
10. 

Here we find that the parts of speech could have been different if the words around it were structured differently. We find a healthy amount of verbs to nouns, and the analysis accurately reflected what we understand when naturally reading the sentences.

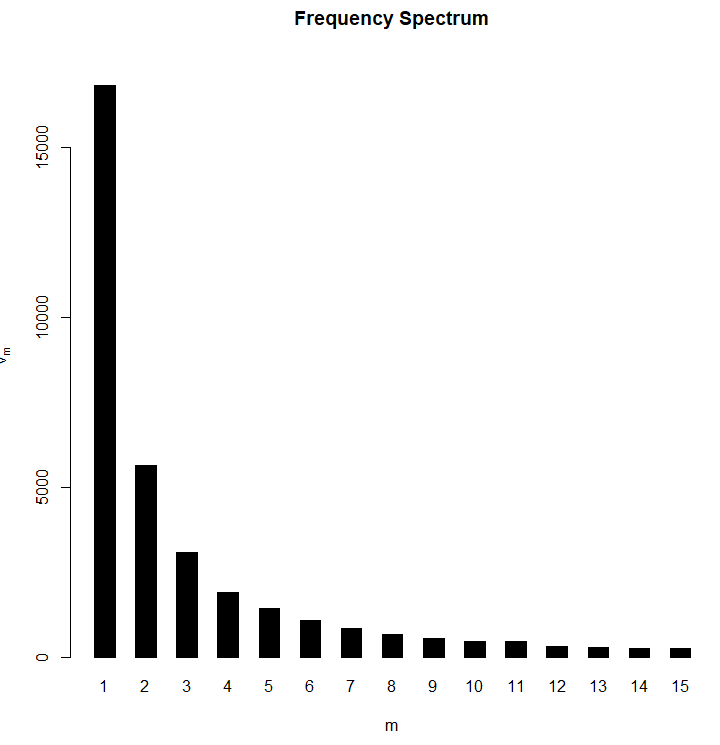
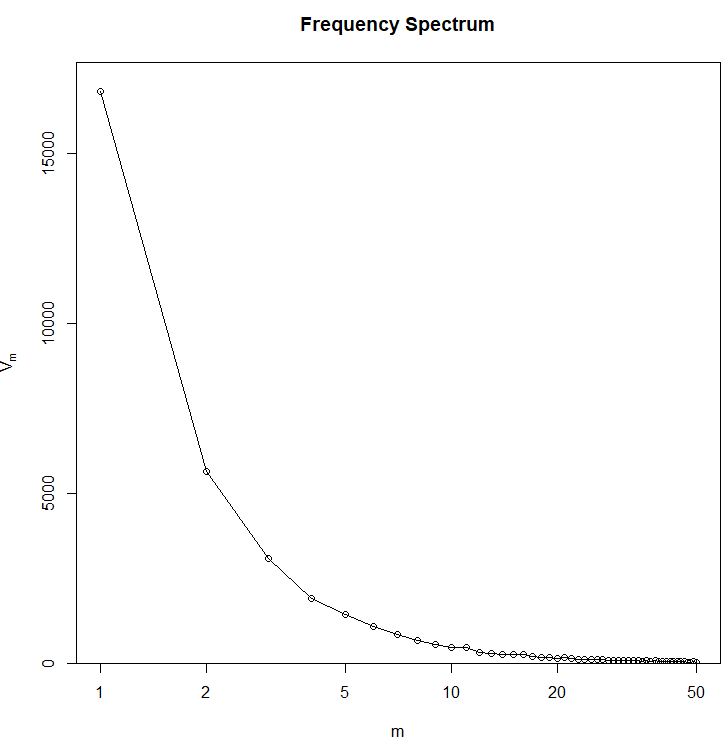
e. According to Zipf’s law, the frequency of any word used in any corpus is the frequency of the most used word divided by it’s rank. This means that the second most used word will be used ½ as many times as the most used word, the 3rd 1/3 as many times, and so on. Using plot with parameters from the zipfR library, such as ItaRi and the spc function, we were able to generate a zipf distribution that accurately displays this property.



This is made using the ItaRi data set. If we take the log on the x axis, we achieve this result:

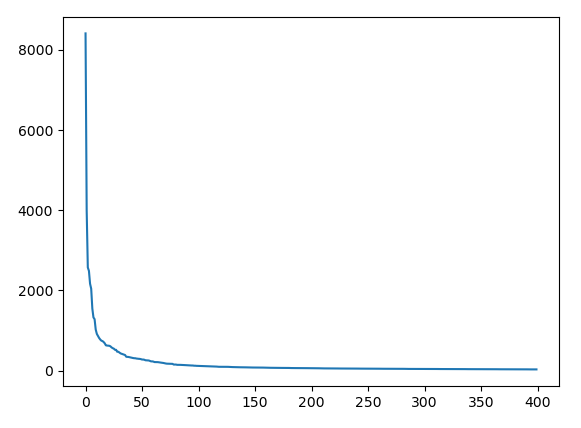


This can also be demonstrated with the BrownInform frequency set:

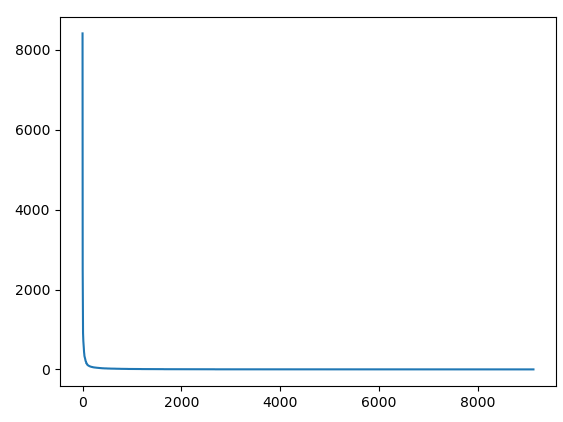
 

Notice that the zipf curves on the BrownInform set are much smoother and follow the law more closely. Not all data sets are this strictly adherent to the rule, but they can all be interpolated close enough to accurately demonstrate the same properties.

When running this through our own program, we wanted to see if TwentyThousandLeagues.txt would also follow this distribution. As shown below, this is clearly indicated:

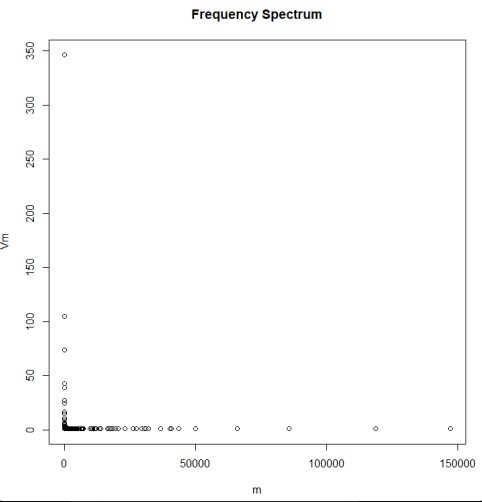


This sample was used on a slice of the data set to make the visualization nicer, but if the entire set it used, we get this distribution:

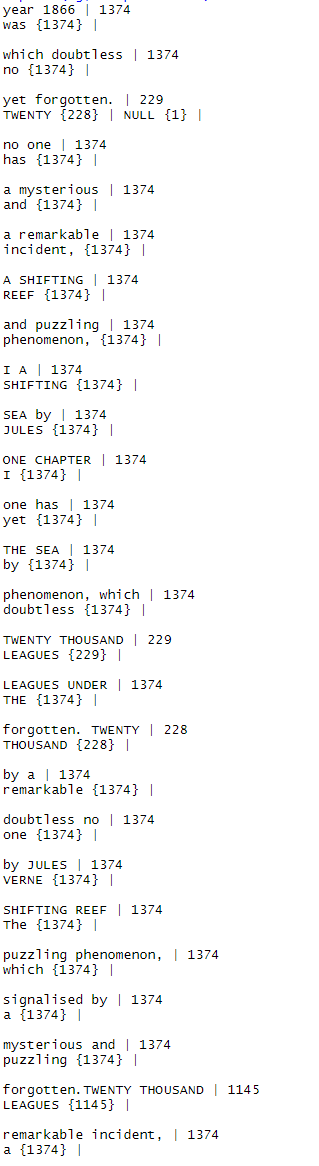


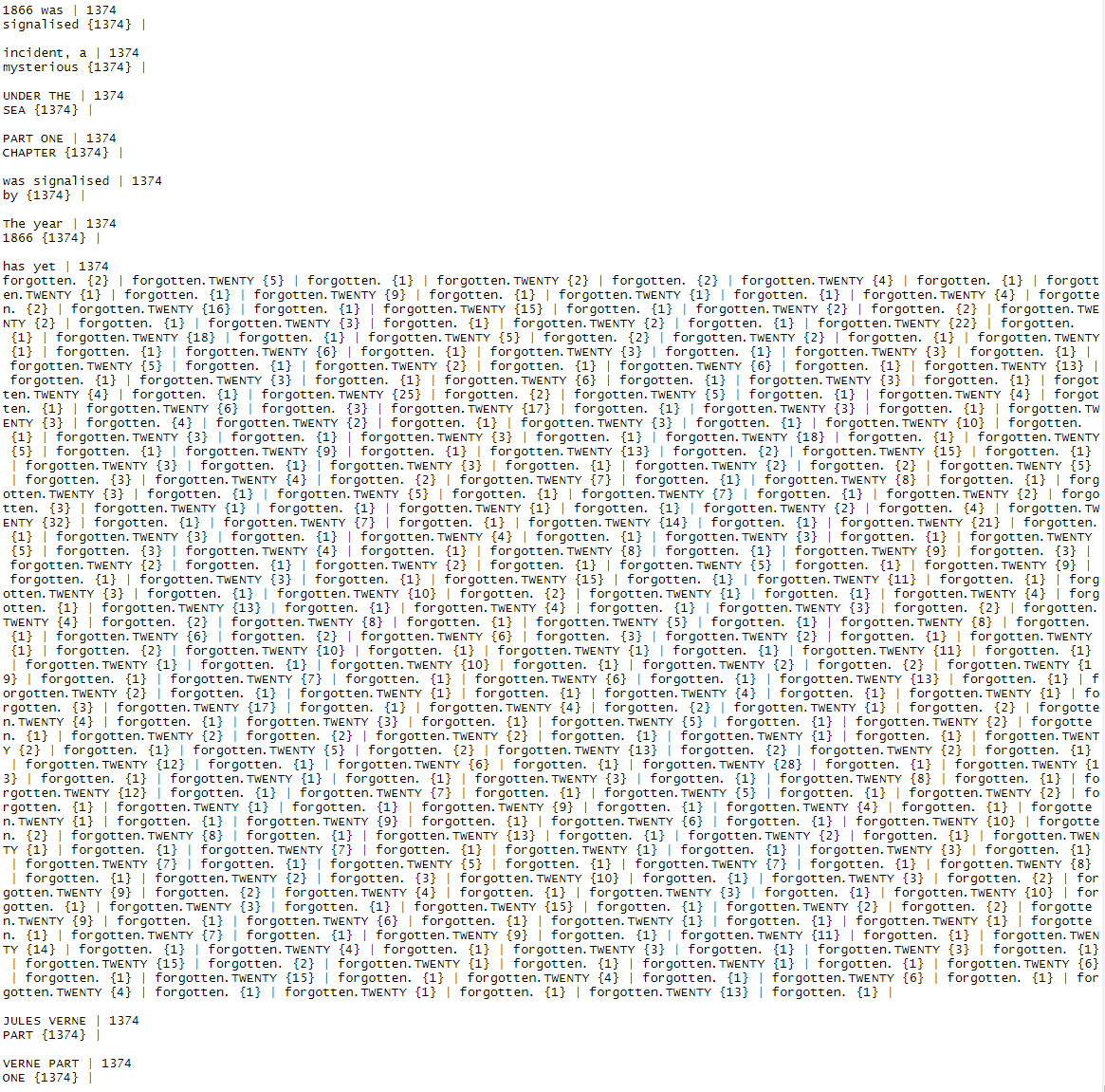
This is the full text, and we find that as we approach lesser used words, we find and increasing number of words that are only used a single time.

Using the zipfR library, we find that this is also true for other texts:

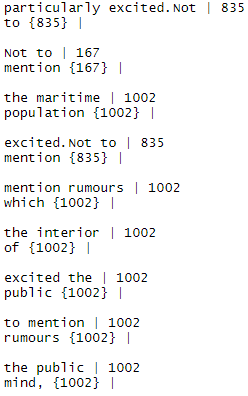


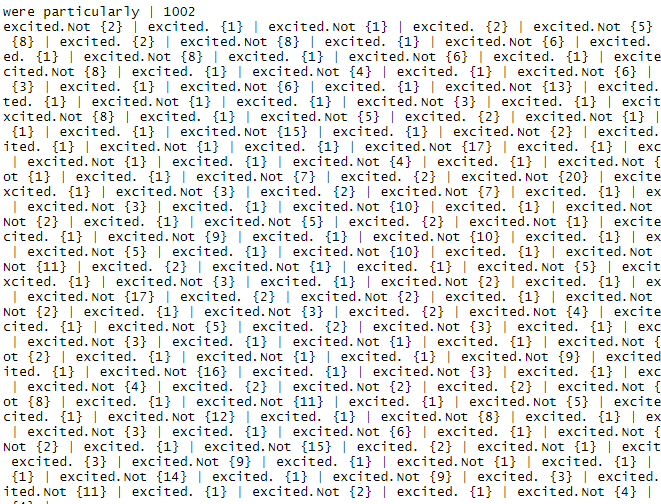
This result is fascinating, and universal through all text and all languages.

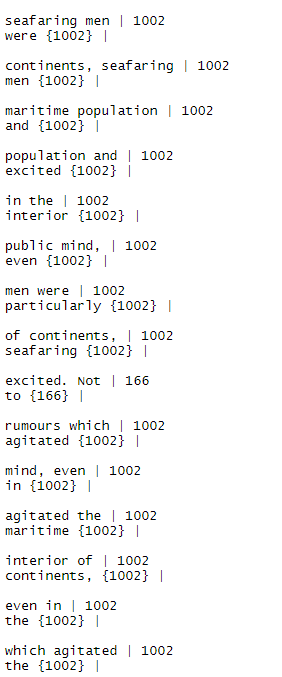
f. sentence 1 



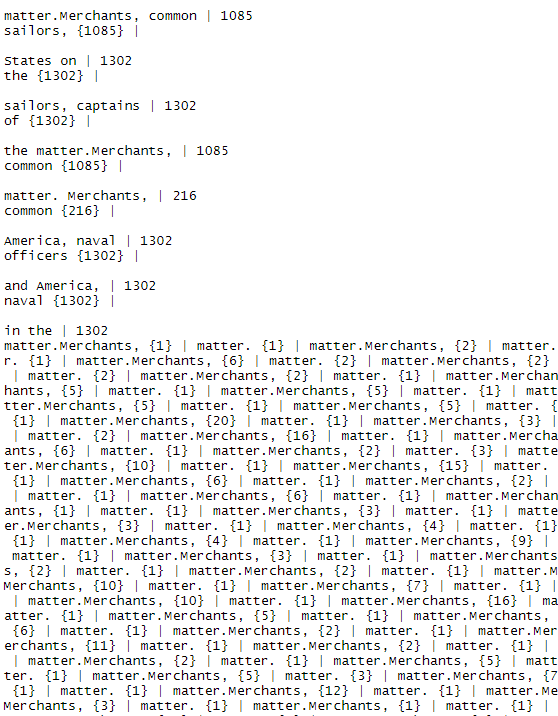
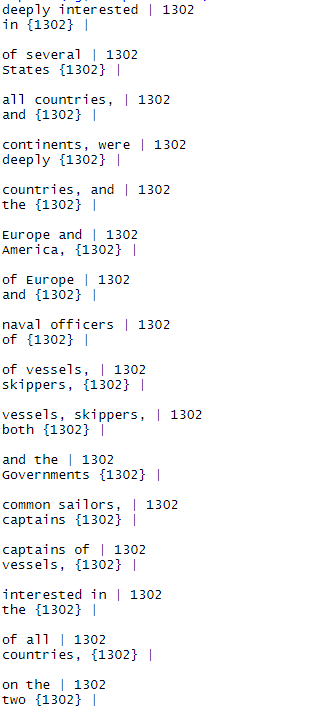
Sentence 2

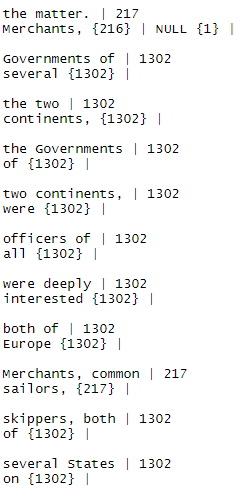




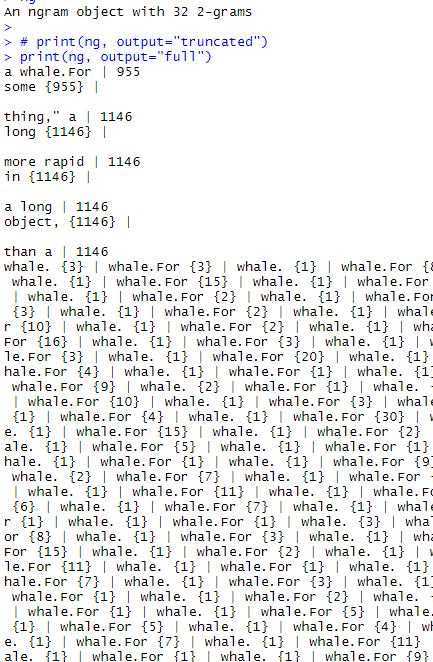


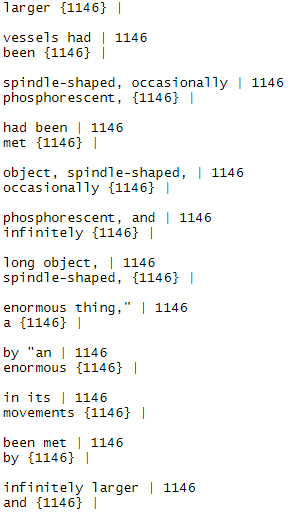
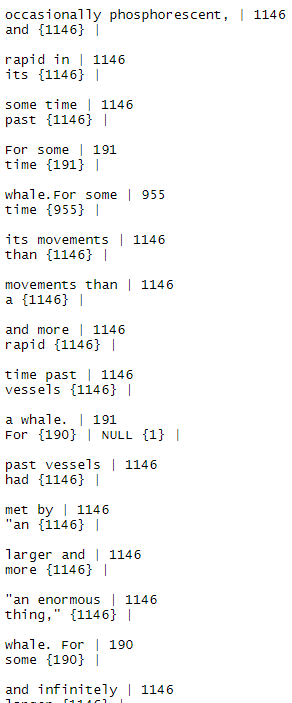
Sentence 3



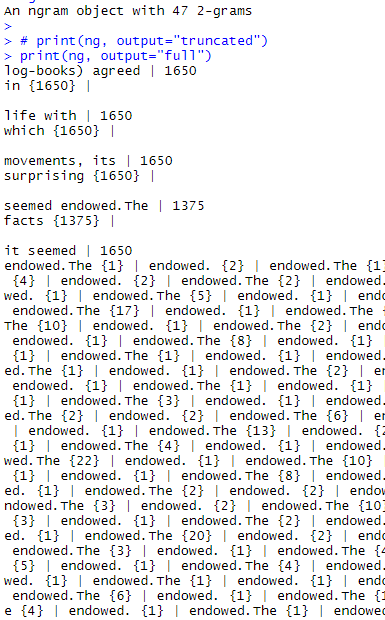
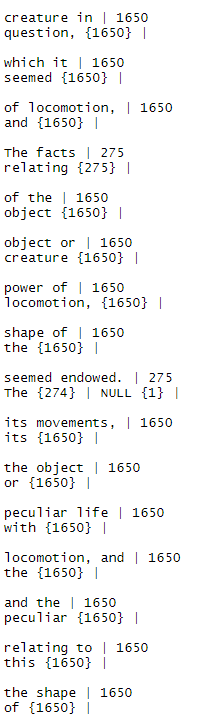


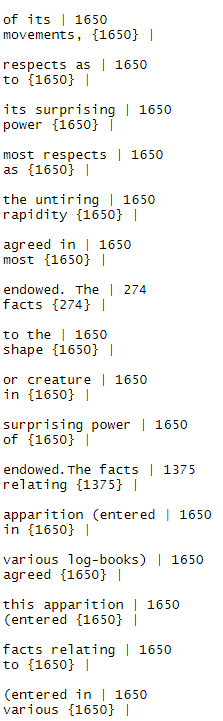
Sentence 4



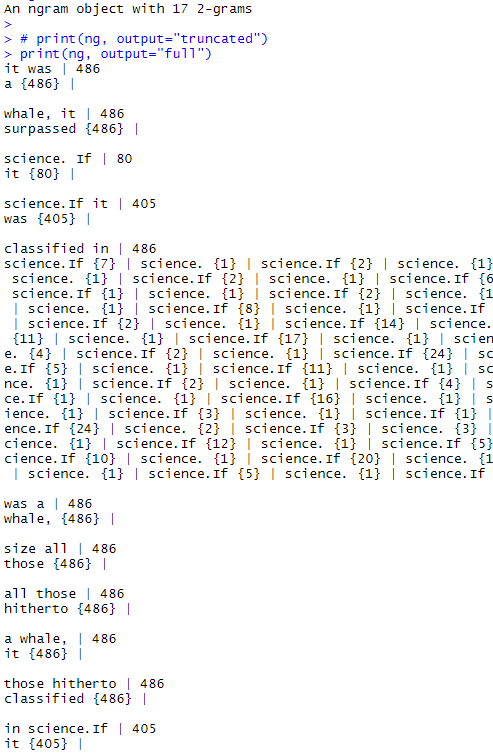
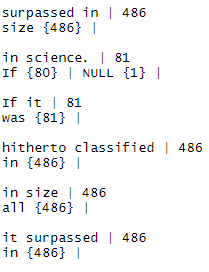


Sentence 5

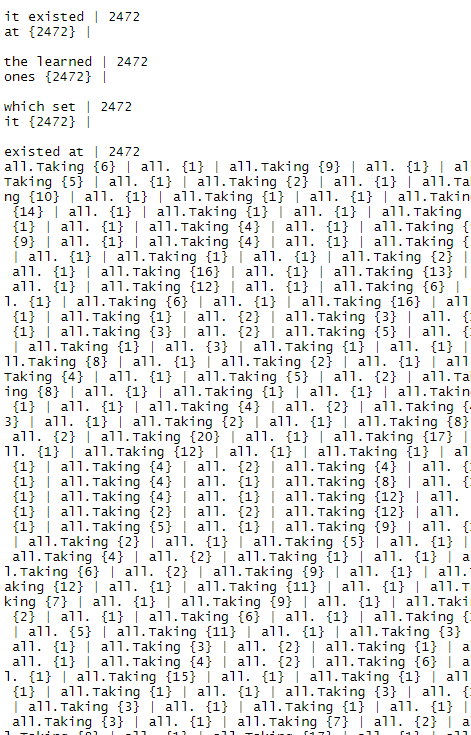
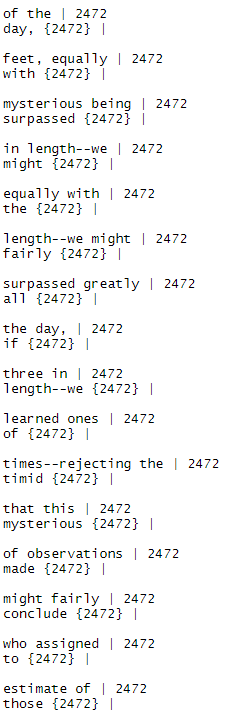
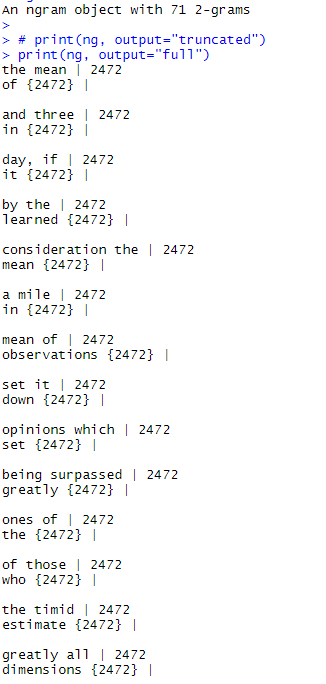
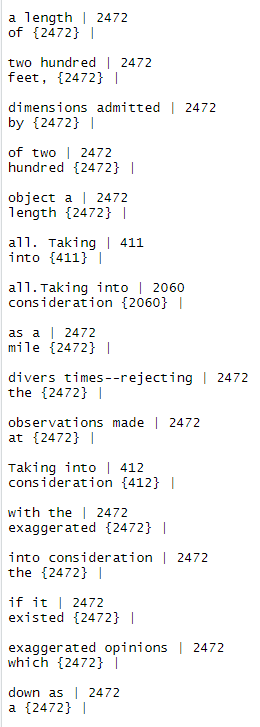
 

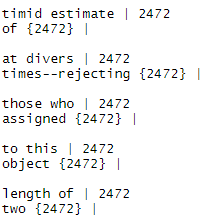
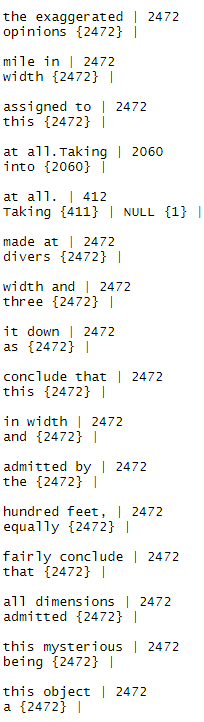


Sentence 6

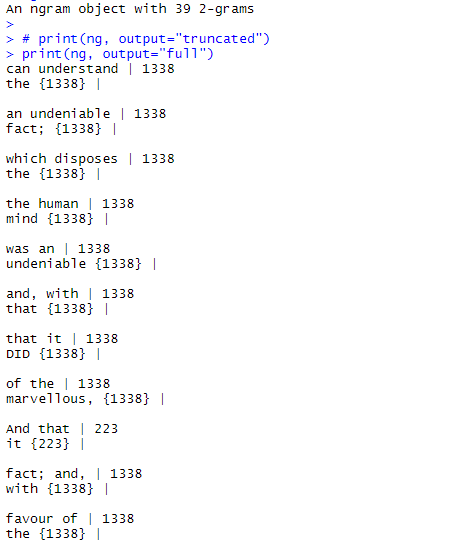
 

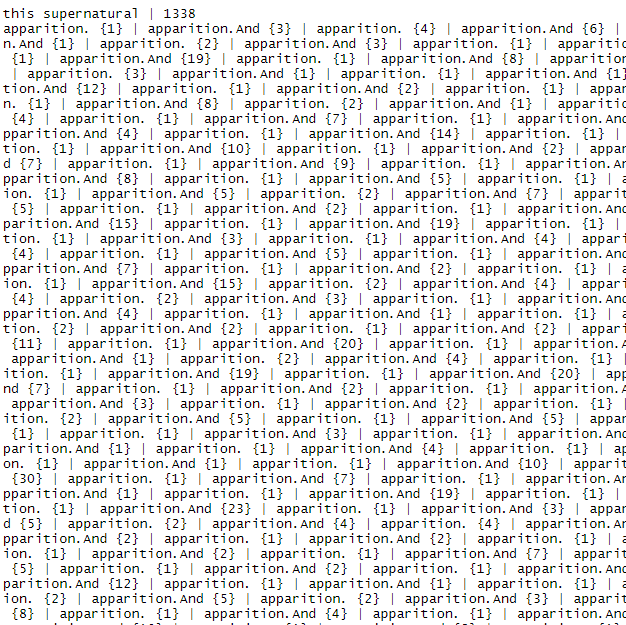
Sentence seven

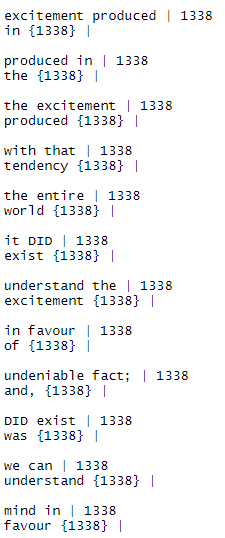
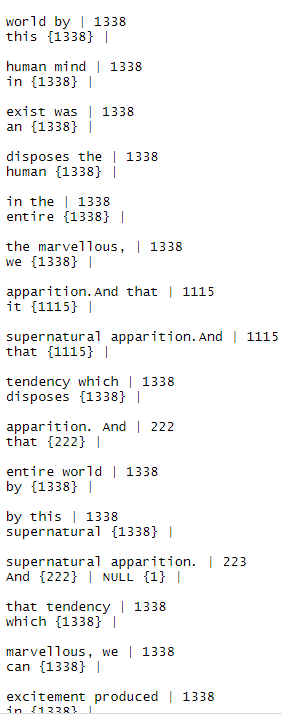
 



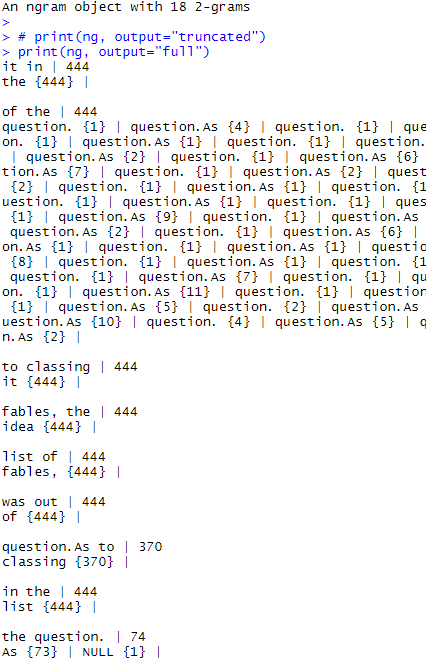
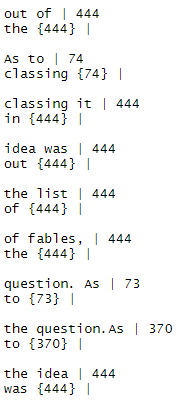
Sentence 8



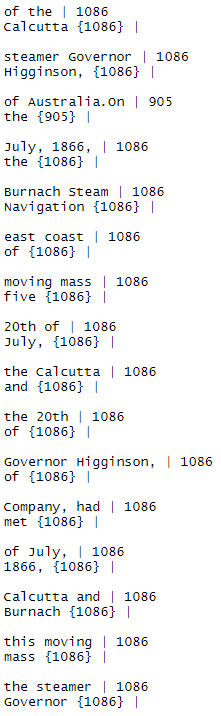


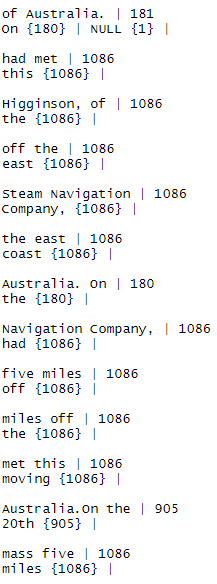


Sentence 9

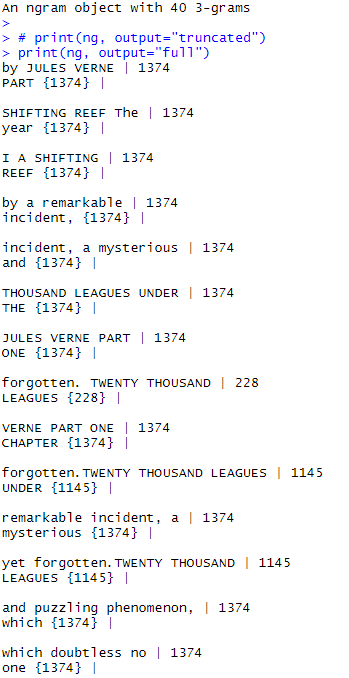
Sentence 10

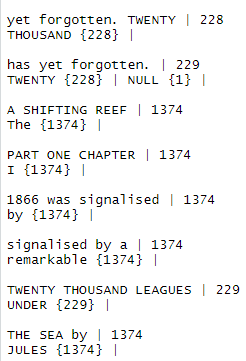
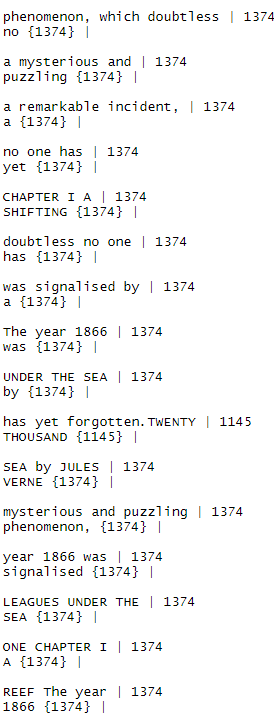
 



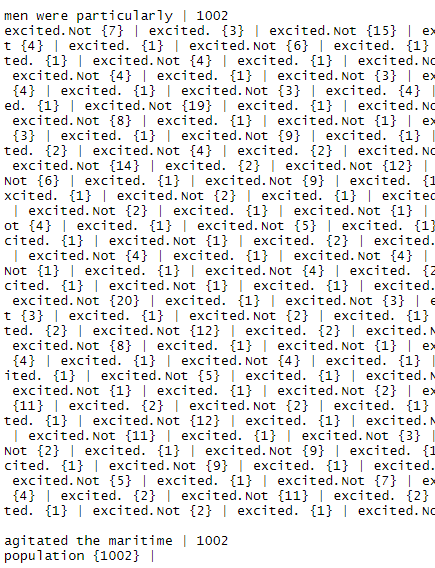
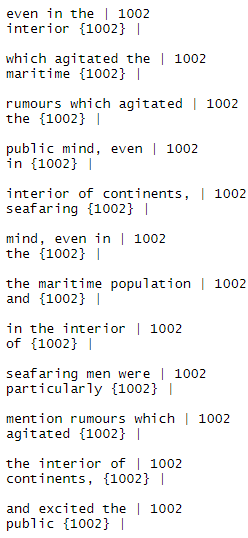
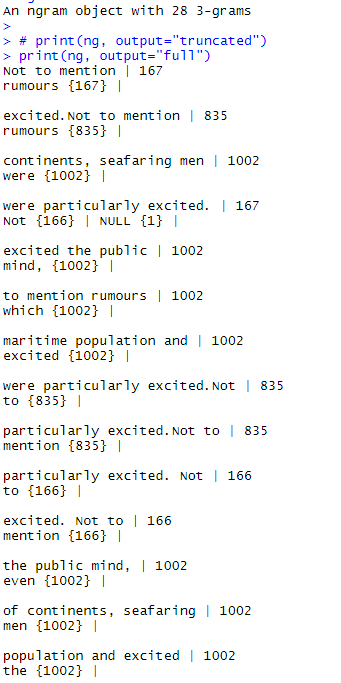
Trigrams:

Sentence 1

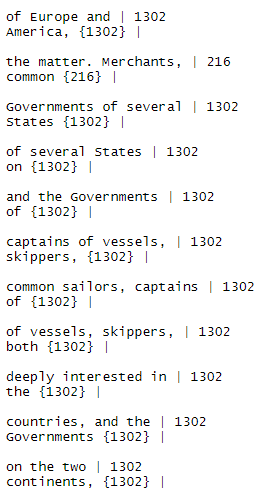
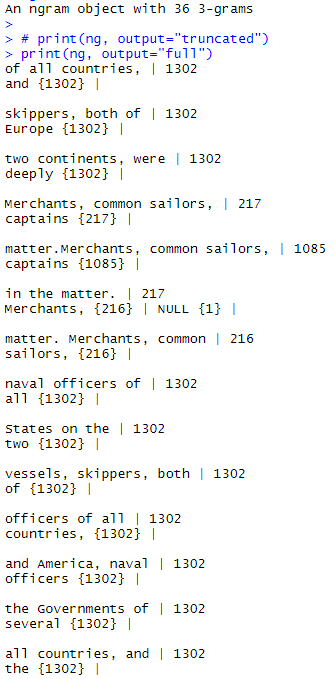


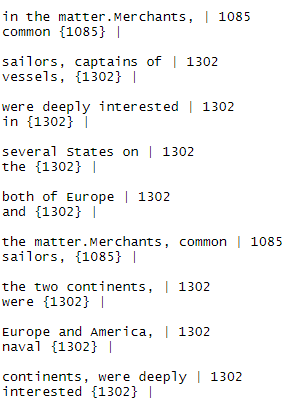

Sentence 2

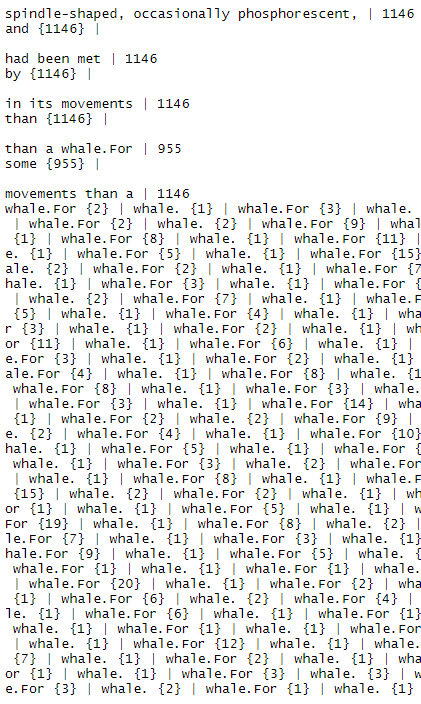
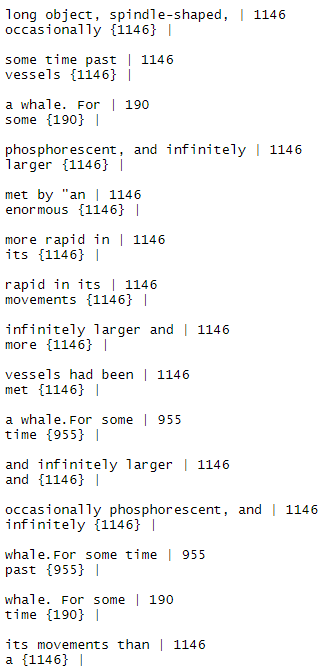
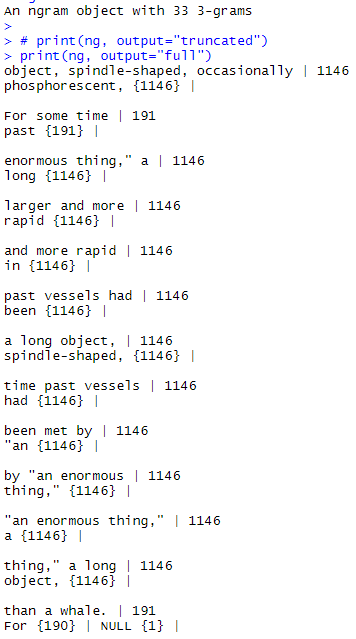


Sentences 3

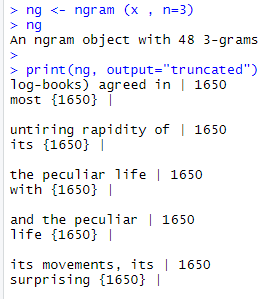




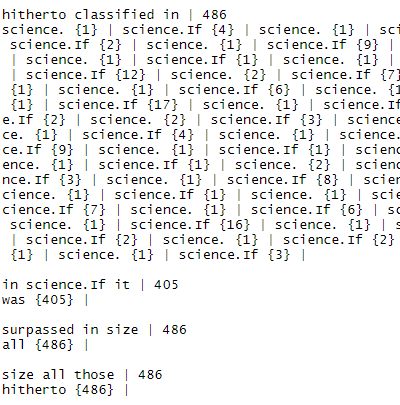
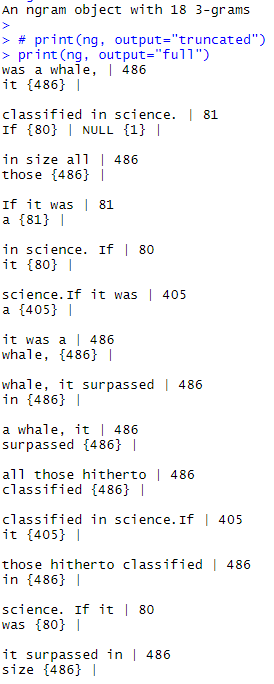


Sentence 4 

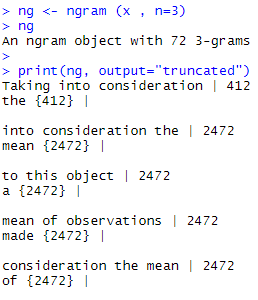
Sentence 5



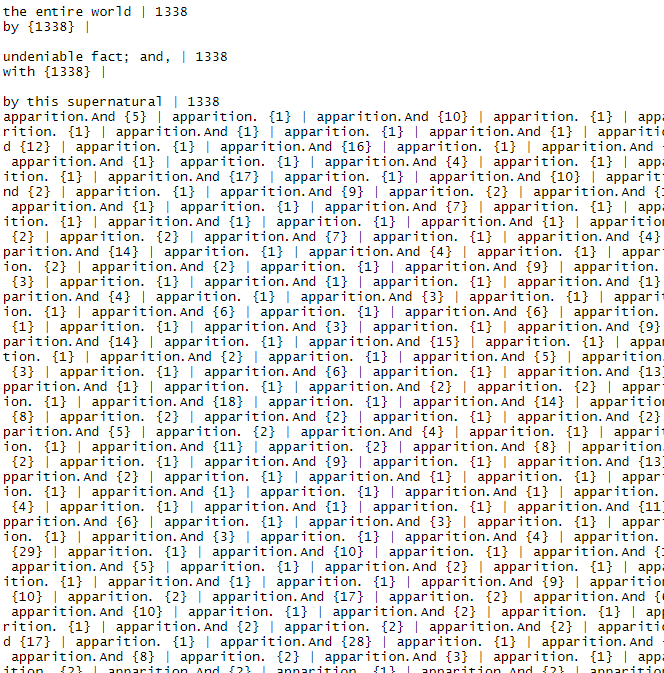
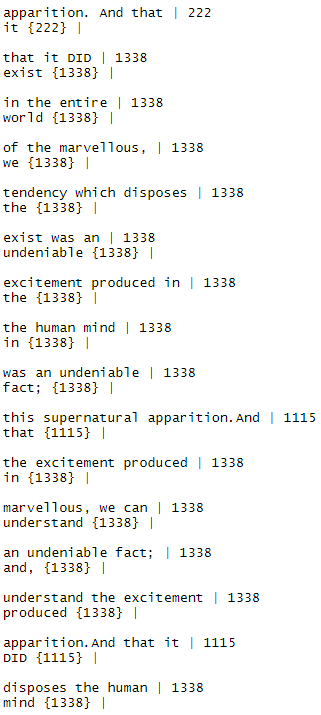
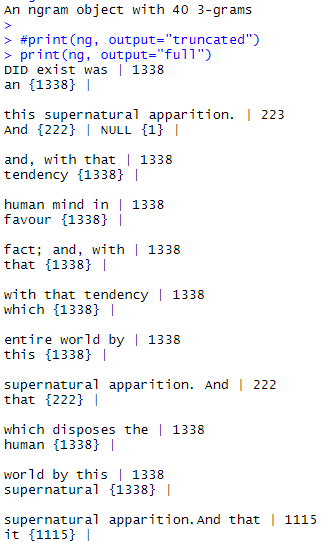
Sentence 6

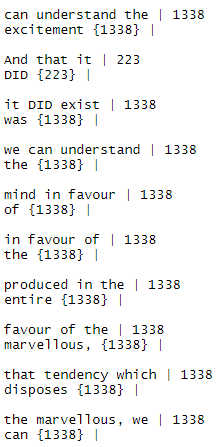


Sentence 7

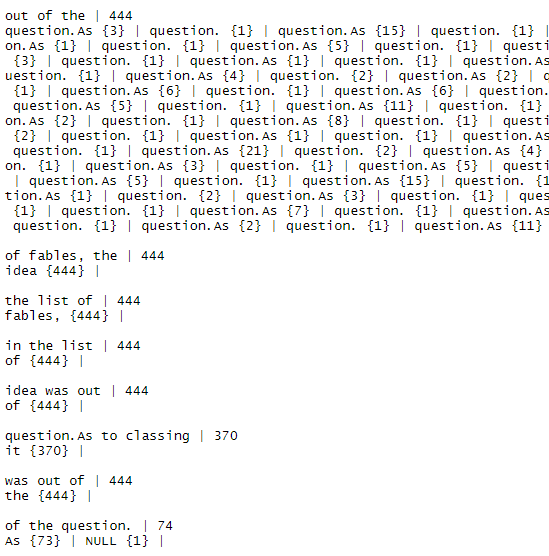
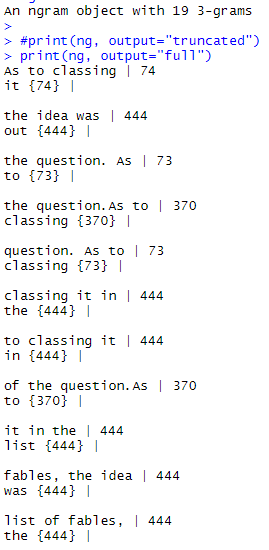


Sentence 8

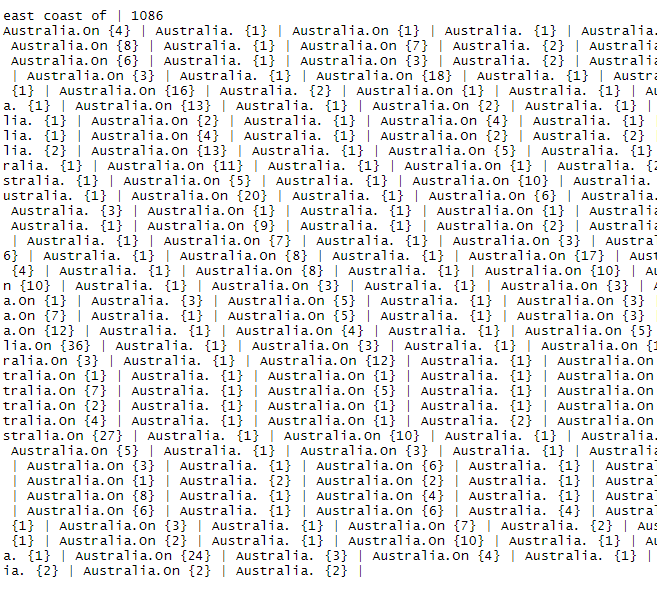
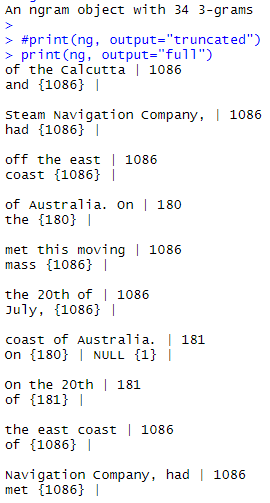


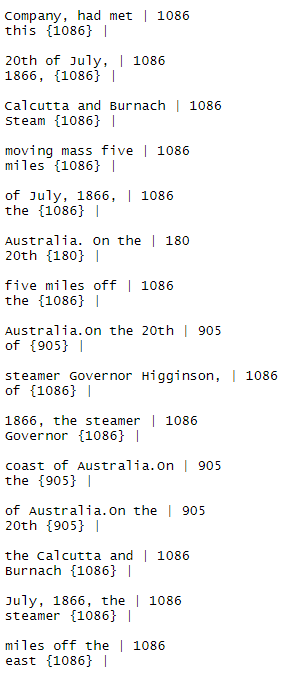
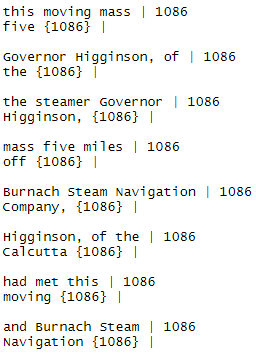


Sentence 9



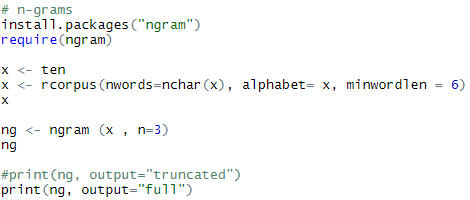
Sentence 10



Analysis:

The bigram/trigram analysis was done using the ngram library with the code below:



A minimum word length of 6 was established, but after analyzing the results, it doesn’t make much sense to exclude words less than this because there are key important pairs of words that appear in the document more than once that should be noted. To exclude any words from these sentences would create bigrams and trigrams that wouldn’t make contextual sense and therefore wouldn’t be useful in any way. Each record in the bigrams and trigrams above have 2 or 3 words that are found together, as well as the word that came after these pairings.

What I learned from d-f

One of the most interesting things I learned about this section was the zipf’s law analysis. It’s very fascinating to see that this simple property can help us analyze other languages that haven’t been decoded yet, and give us small insights into how their alphabets look. This mathematical property allows us to look into the past to uncover universal similarities between cultures. Another interesting piece here is the parts of speech tagging. One of the key components to analyzing speech and text is tokenization, which is what this strategy does. This allows for easier analysis of language and intention, and I had not previously worked with libraries that did this in the past other than programming language compilers.

R functions used:

require(openNLP) <- library

Maxent\_Sent\_Token\_Annotator()

Maxent\_Word\_Token\_Annotator()

annotate(text, list(sent\_token\_annotator, word\_token\_annotator))

require(ngram) <- library

rcorpus(nwords=nchar(x), alphabet= x, minwordlen = 6)

ngram (x , n=3)

require(zipfR) <- library

paste(c("he wants to back", collapse= ""))

as.String(text)

ItaRi.spc

summary(ItaRi.spc)

with(ItaRi.spc, plot(m, Vm, main="Frequency Spectrum"))

plot(BrownInform.spc)