Text Analyzer: Categorizing User Input

STAT 385 SP2019 - Team Random Group 5

Erik Caster (caster2)
James Crouse (jcrouse2)
Hyunsoo Lee (hyunsoo2)
Daniel Stumpf (djs2)

May 9, 2019

Abstract

For the project, our group decided to build a shiny application that will analyze text documents and give a probability model of the language, genre, and era of the document. Users will be able to see the probability of each category evaluated from built models. This will allow the user get a sense of the content of the document and decide whether they want to read it. From this, we hope to recognize different modeling and analyzing methods, along with the ability to display the information inside text effectively.

Contents

1 Introduction

Through the book *Brave New World*, Aldous Huxley has shown the world without much restriction but full of distraction (Postman 2006). With the rise of the internet, this description accurately fits current times. In the era when data is easy to be shared and collected, natural language processing is taking an increasing role in extracting relevant information (Jurafsky and Martin 2009). For example, a google search function allows the user to disregard websites containing certain phrases by using "-word" keyword. This saves so much time for the user since there are a lot of homographs and information that will be extraneous.

This project focuses on natural language processing, specifically on using it to analyze text files into three different categories: language, era, and sentiment. This will save the user's time in determining the content of the text and deciding on whether they will enjoy it or not. The language of the text may seem obvious to the user if the user is fluent in that language. However, the application provides more analysis than that. We will authorize the user to choose different modeling methods, allowing the user to view the accuracy of each method. This would also provide them with how efficient and accurate each method will be. Human-computer interaction is an important field in Linguistics, and both user and our group would be able to determine the best model and improve the training dataset or the model. The model will be implemented using different statistical methods such as conditional probability using N-grams, tagging, and more.

2 Related Work

There has been a lot of research done on different ways to split and analyze text. We will be employing different methods to make the model as accurate as possible. Our program will be unique in the sense that it develops many different language processing models and allows the users to test and choose the most efficient technique they want to apply. We looked at different methods we wanted to implement in this project.

The first method the project will be applying is N-grams. An N-gram probability is computed by taking the conditional probability of a word given previous N-1 words. The advantage of N-grams is that they get access to great amount of lexical knowledge. The disadvantage of it is the fact that they are very dependent on the trained data (Jurafsky and Martin 2009).

Another method that we will be implementing is tagging. Tagging entails using three different methods, AFINN, bing, and NRC. Each of these methods provides different semantic scores for whether it is a positive or negative sentiment. AFINN assigns words a score between negative five and five, based on how negative or positive the word is. Bing assigns words scores in a binary fashion, either being listed as positive or negative. Lastly, NRC labels the word on a specific emotions, based on a preset list of possibilities (Silge and Robinson 2017).

We will first attempt tagging one word at a time to analyze our training data, while then looking into possible arcs of a book based on these different sentiments to build a model to predict the genre of a text given. A paper published in the IEEE department at University of Illinois talks about the use of sentiment analysis on web-scraped data to predict the opinions over opinionated pieces on the internet (Cambria et al. 2013). While the paper attempts to use sentiment analysis to focus more on brand awareness or popularity, we will be using the analysis to find the genre of a book created by models using the famous books that practically defined genres as we know them today. Also, we will be utilizing this to help distinguish similar language patterns used in texts.

The use of word cloud is not a new idea, it has been implemented countless times and is a staple for word frequency analysis. One such example was performed by Katie Swan in her analysis of 6 specific book's word frequencies in relation to determining the time period of the text, the style of the author, and the subject of the book (Swan n.d.). Using word cloud, Swan separated the most common positive and negative words into a gradient. Swan paired this with bar graphs denoting the connotations of words, dividing them into different sentiments. Similar to Swan's analysis, our project will be using a word cloud and a frequency dictionary. However, we are also employing other factors to improve the model's accuracy.

The Text Mining with R document has provided us with a good background on tidy text. The document helps explain `tidytext` format and the uses of `tidytext` package. While talking about the format, the document introduces the ideas of tokens, the gutenberg r package, word frequencies and summaries. It also brings up sentiment analysis with tidy data and how to use wordcloud with `tidytext` data. The document also brings up analyzing word and document frequencies along with n-grams and correlations. This document serves as a good introductory information piece that allows us to dive deeper into investigating text analysis with R (Silge and Robinson 2017).

3 Methods

The data for training were text files in each category that we chose. For example, we looked at taking *Pride and Prejudice* as English Romance Novel from Romanticism Era. We got the text file from the Project Gutenberg website. Accessing the book files should not be an issue, as there is already a function available which allow us to read famous works as text files into R (Refer to Table 1 in the Appendix). We only copied ".rds" files and "global.R", "server.R", and "ui.R" files in the repository. In order to see the whole processing step, refer to this github repository and take a look at "preprocess-data.R" file.

In order to create the model and analyze a given text, the string needs to be tokenized. The library used for tokenization was function called `unnest_tokens` in a package called `tidytext`, which allows quick tokenization of words. It also includes a good subset of stop words that allow easy filtering of tokens (Silge and Robinson 2017). However, I ended up using stop words from the `tm` pacakge since it contained stop words for other languages as well (Feinerer and Hornik 2018). By using these methods that allow tokenization, we then used the `dpylr` package to filter through our now tokenized data, which helped us to summarize our findings (Welbers, Atteveldt, and Benoit 2017).

We then implemented different modeling methods for each category. For determining language, we will looked at n-grams. We implemented bigrams, trigrams, quadgrams. It was done utilzing the ngram-tokenizers function in `tidytext` (Silge and Robinson 2017). Then, we implemented different smoothing methods: laplace and good turing's smoothing using R functions.

For the genre, the group used `dplyr`'s functions under the join category, to help compare our text file to the different scoring methods (Wickham et al. 2018). These different methods include NRC, AFINN, and Bing. Each of these data sets allowed us to use the different join commands to help us score our different texts with different sentiments. With these new tokenized words and scores, we plan to train the model based on the frequency of words, or different net scores across pages and chapters to help predict the book's sentiment (Silge and Robinson 2017).

To build the model, we assigned the defined genre for each book. Much of this analysis came from the use of the function LDA inside the `topicmodels` package which allows for comparison of texts different frequency of a specific word (Grün and Hornik 2011). With this, we were able to compare the similarity of the word usage in the given book with the usages in other books across our database.

For our visualizations, we used the `ggplot2` package and used the geom_bar function to create a bar graph that compares the probabilities of if the text is a specific language, era, or genre based on our model's output (Wickham 2016). We also used tm_map function inside `tm` to create a word cloud looking at its word frequency (Feinerer and Hornik 2018). We looked to find the best visualization method for frequency.

4 Results

From the shiny application we can see the results of language modeling, era modeling and sentiment analysis. From the model on language modeling we can see that the language prediction is given by the lowest log probabiity in both laplace smoothing and Godd Turings methods. We also see that we can change the language model using bigram, trigram and quadgram models. (Refer to Figure 1 on the Appendix).

From the era modeling we can see that the model returns scores for each era, highlighting the biggest one. Indicating that it is from the era. Also the model displays each time go along with the years of each era. Changing the language will could change the era the text is said to be from, so users must be able to corretly choose language. (Refer to Figure 2 on the Appendix).

From the sentiment analysis model we can see the frequencies in repect to positive connotation while it is grouped by posititvity. It also provides us with a wordcloud, colorized by sentiment.(Refer to Figure 4 on the Appendix). Once we change the senitment analysis from positive to emotion, we are given a new frequency table with more detailed types of sentiment. (Refer to Figure 3 on the Appendix).

5 Discussion

The goal of the project was to be able to help the user be able to quickly analyze a piece of text and see if they wanted to contiune on reading the text. From the results of the shiny app, users are able to see the language, era, genre and sentiment of the text quickly. This gives the user a broad range of information on the text, helping them decide to read the text or not. By showing the results of the analysis, we have help the user quickly analyze the text which was our primary goal.

6 Conclusion

In the world full of data, information overload easily occurs. For this project, we are focusing on creating an efficient way to sort and extracts information of texts, dividing them into three different categories: language, time period, and genre. Using the packages like `tidytext`, `ggplot2`, and `dyplr` along with the gutenberg text library, our group will be able to train an application to analyze texts and produce simple yet pertinent visuals based on the word frequency and the sentiment analysis.

7 Appendix

Table 1: This is the sample data retrieved using the Gutenberg Project website

gutenberg_id	text
1514	A MIDSUMMER NIGHT'S DREAM
1514	
1514	by William Shakespeare
1514	
1514	
1514	
1514	
1514	
1514	Persons Represented.
1514	
1514	THESEUS, Duke of Athens.
1514	EGEUS, Father to Hermia.
1514	LYSANDER, in love with Hermia.
1514	DEMETRIUS, in love with Hermia.
1514	PHILOSTRATE, Master of the Revels to Theseus.
1514	QUINCE, the Carpenter.
1514	SNUG, the Joiner.
1514	BOTTOM, the Weaver.
1514	FLUTE, the Bellows-mender.
1514	SNOUT, the Tinker.
1514	STARVELING, the Tailor.

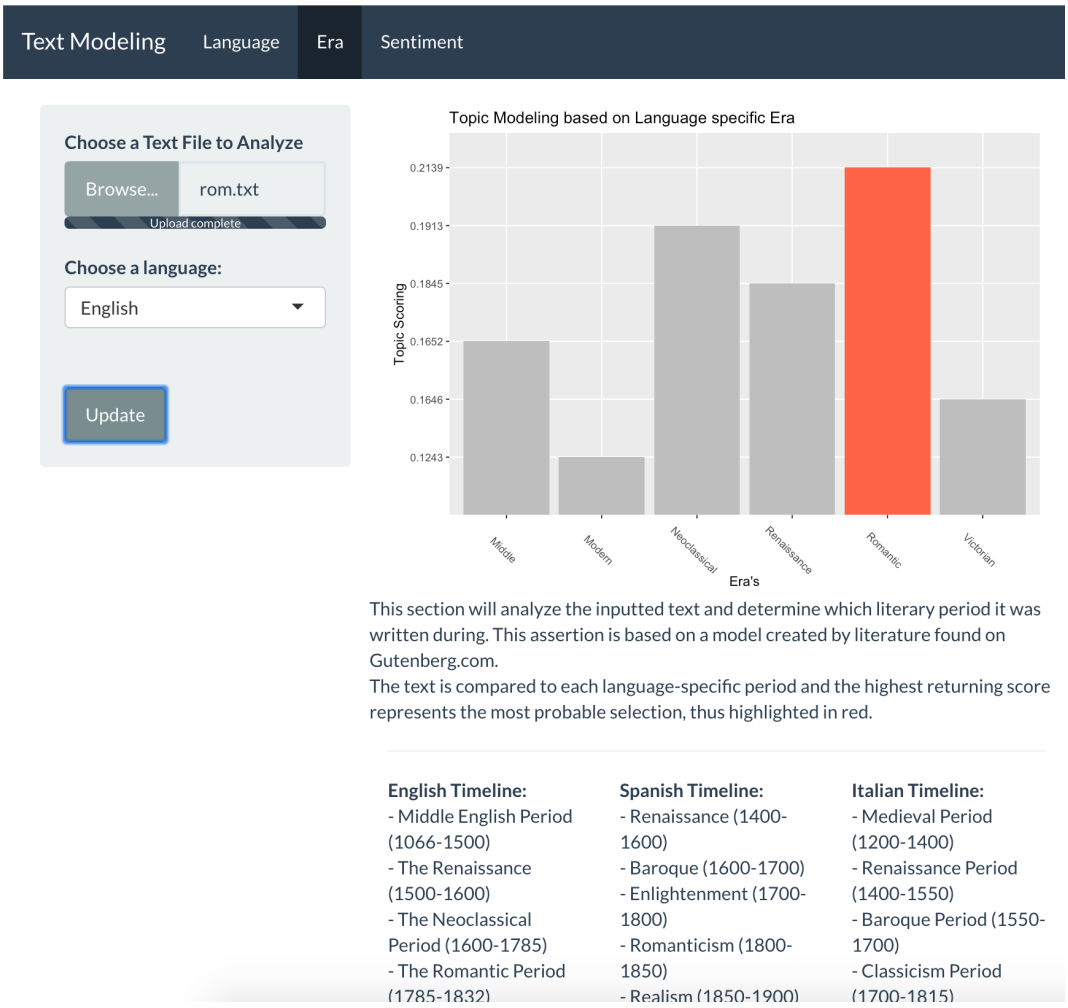

Figure 1: Language Model Analysis

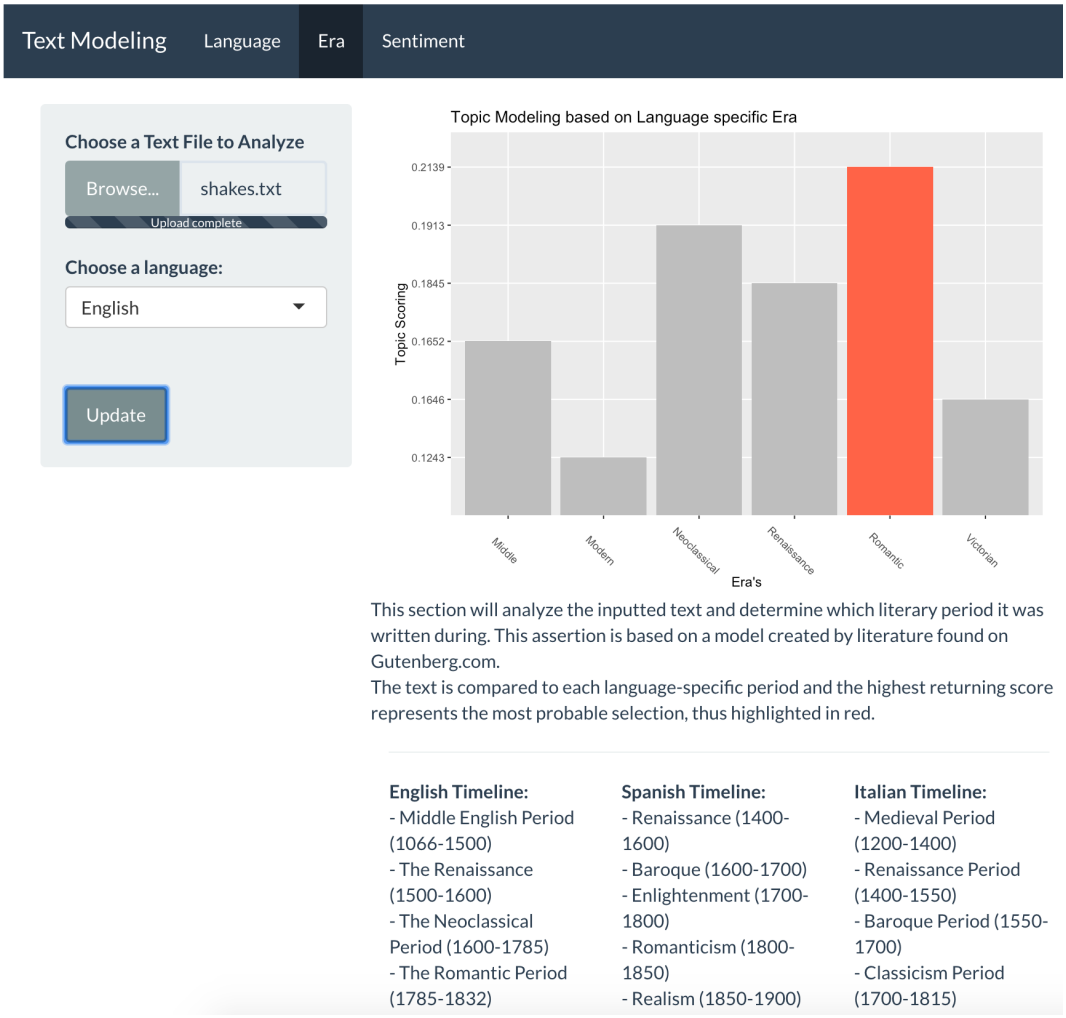

Figure 2: Era Model Analysis

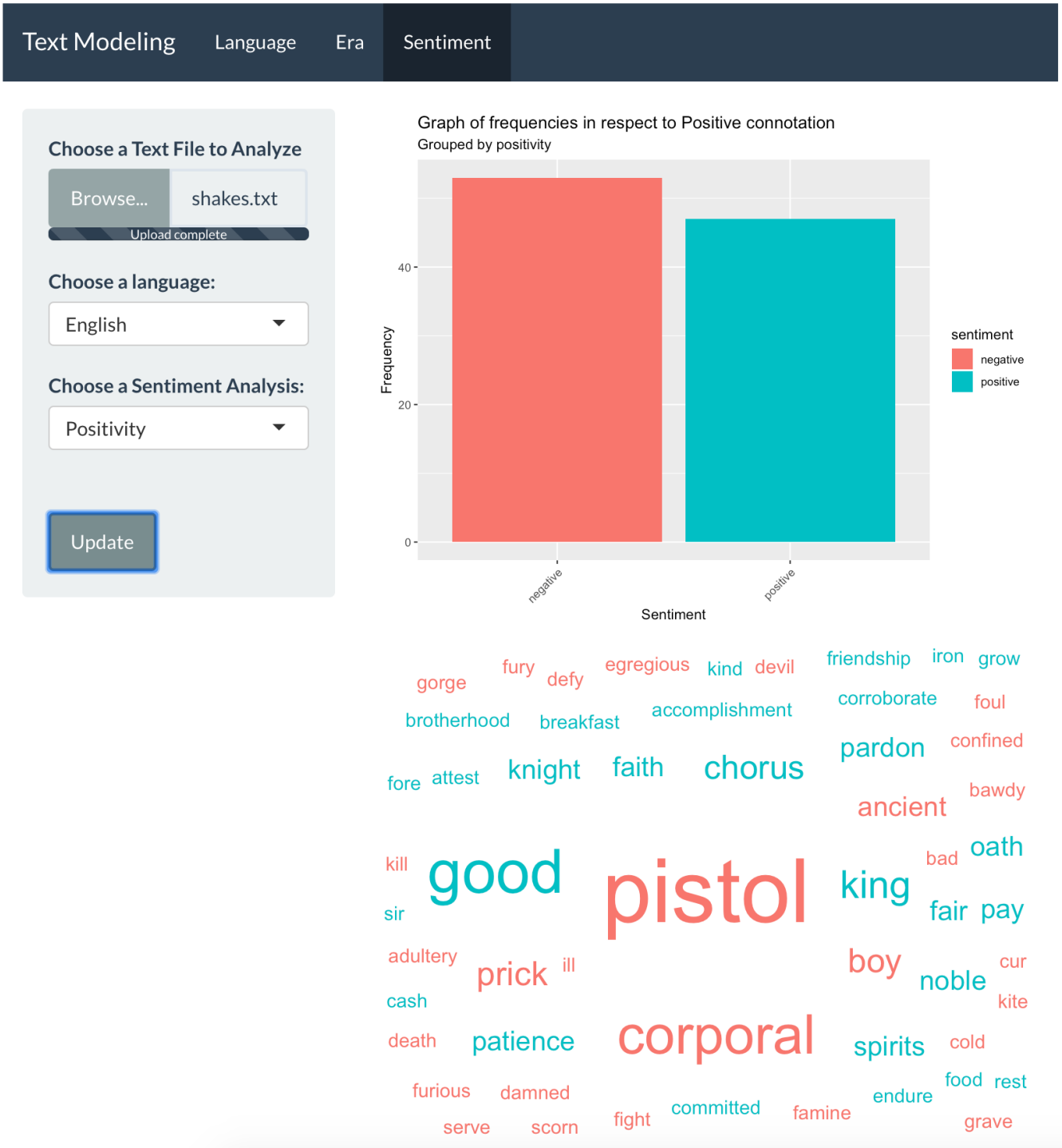

Figure 3: Sentiment Analysis Model with positivity

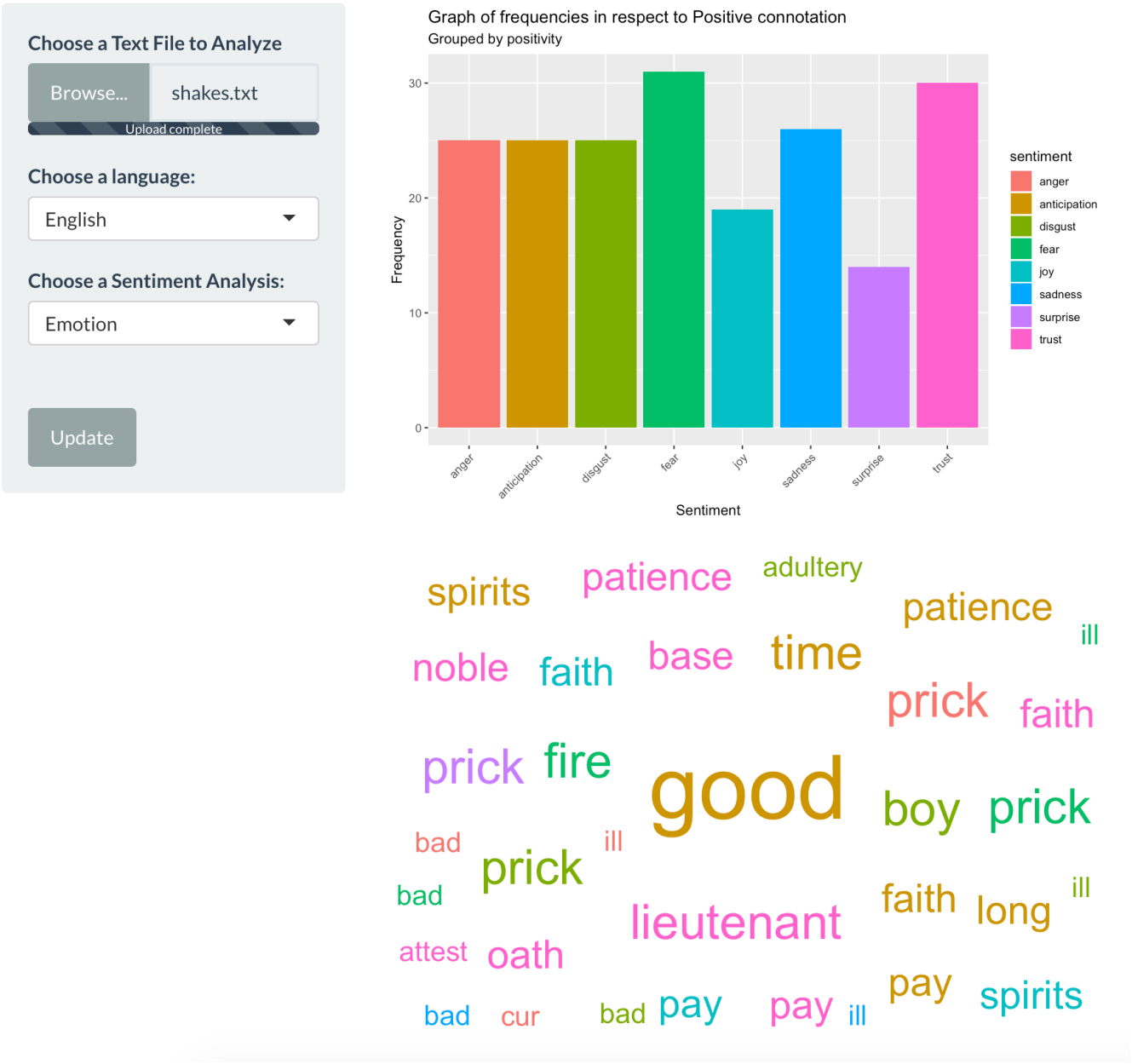

Figure 4: Sentiment Analysis Model with emotion

References

Cambria, E., B. Schuller, Y. Xia, and C. Havasi. 2013. "New Avenues in Opinion Mining and Sentiment Analysis." *IEEE Intelligent Systems* 28 (2): 15–21. https://doi.org/10.1109/MIS.2013.30.

Feinerer, Ingo, and Kurt Hornik. 2018. *Tm: Text Mining Package*. https://CRAN.R-project.org/package=tm.

Grün, Bettina, and Kurt Hornik. 2011. "topicmodels: An R Package for Fitting Topic Models." *Journal of Statistical Software* 40 (13): 1–30. https://doi.org/10.18637/jss.v040.i13.

Jurafsky, Daniel, and James H. Martin. 2009. *Speech and Language Processing (2Nd Edition)*. Upper Saddle River, NJ, USA: Prentice-Hall, Inc.

Postman, Neil. 2006. *Amusing Ourselves to Death: Public Discourse in the Age of Show Business*. Penguin Books. https://books.google.com/books?id=zGkhbPEjkRoC.

Silge, Julia, and David Robinson. 2017. *Text Mining with R: A Tidy Approach*. Sebastopol, CA: OReilly Media.

Swan, Katie. n.d. "Analysis of Sentiment and Word Frequency in Six Novels." Accessed March 14, 2019. http://student.elon.edu/kswan4/finalproject/.

Welbers, Kasper, Wouter Van Atteveldt, and Kenneth Benoit. 2017. "Text Analysis in R." *Communication Methods and Measures* 11 (4). Routledge: 245–65. https://doi.org/10.1080/19312458.2017.1387238.

Wickham, Hadley. 2016. *Ggplot2: Elegant Graphics for Data Analysis*. Springer-Verlag New York. http://ggplot2.org.

Wickham, Hadley, Romain François, Lionel Henry, and Kirill Müller. 2018. *Dplyr: A Grammar of Data Manipulation*. https://CRAN.R-project.org/package=dplyr.