
Introduction to Text Analysis: A Coursebook
Brandon Walsh and Sarah Horowitz

# Table of Contents

3. Adapting This Book 1.14.3

# Preface

# Preface

*(Note: We welcome feedback on this book! If you find an error, want clarification on a particular issue, or find deep problems with particular explanations, drop us a line on our GitHub issues page. We'll be grateful and list you in our acknowledgements!)*

This workbook provides a brief introduction to digital text analysis through a series of three-part units. Each unit introduces students to a concept, a tool for or method of digital text analysis, and a series of exercises for practicing the new skills. In some cases, studies of particular projects are presented instead of tools in the third section of each unit.

The materials here are meant to form the basis for a digital text analysis course that does not require extensive training in programming and is written with student readers in mind. Originally developed for use in a course titled "Scandal, Crime, and Spectacle in the Nineteenth Century," this workbook draws from these course materials for its datasets and prompts. The book is intended to be modularized enough that it could be used in conjunction with other courses either in whole or in part, as all of its materials are openly available on GitHub. The tripartite structure of each chapter means that sections can be easily removed and replaced with different tools or content. In particular, we envision our course-specific exercises in the third section of each chapter to be removable. For more guidance on how to remix the work for your own ends, see Adapting This Book.

The book is best viewed online using either Chrome or Firefox. You can also download it to read as a PDF here.

# Acknowledgements

# Acknowledgements

We are very grateful to the encouragement and feeback from all of our colleagues as we put these materials together. In particular, we would like to thank the following individuals for their advice on specific portions of the book:

- Mackenzie Brooks
- Eliza Fox
- Julie Kane
- Eric Rochester
- Demitra Tsioulos
- Students from HIST 211

In addition, the cover image is word cloud generated by Voyant, an excellent tool developed by Stéfan Sinclair and Geoffrey Rockwell that we discuss in our chapters on "Cyborg Readers" and "Reading at Scale."

# Introduction

# Introduction

# For Instructors

# For Instructors

This coursebook provides a brief introduction to digital text analysis through a series of three-part units. Each unit introduces a concept, a tool for digital text analysis (or case studies based on the associated concept), and then provides a series of exercises for practicing the new skills. Our intended audience is students who have no background in programming, text analysis, or digital humanities.

We designed this book with three goals:

First, we wanted to provide materials for a text analysis course that does not require extensive training in programming. Courses in text analysis with R or Python are valuable and have their place, but many concepts in text analysis can be covered through a tools-based approach. In part, this decision was made due to time restrictions. These particular materials developed as companion pieces to the equivalent of a one-credit digital humanities lab for a three-credit history course at Washington and Lee University. Thus, the amount of time available for instruction in digital humanities and programming was minimal. Choosing tools instead of languages, we hoped, would allow for the exploration of more disciplinary material than we might otherwise have time for. Accordingly, here we introduce concepts and methods gradually and over the course of the term. While some of these tools are more difficult to use than others, the book requires minimal prior experience with programming to work through these materials. In the course of the book, however, we introduce basic programming concepts necessary to working with unstructured data in a natural language processing context. If anything, we hope this book will provide a taste of what can be gained from further study that *does* use programming.

Second, we wanted to provide a set of materials that could be resuable in other contexts. In this, we were inspired by Shawn Graham's course workbook on Crafting Digital History. Our own workbook was originally developed a course in nineteenth-century European cultural history, and it draws from these course materials for its datasets and discussions. As much as possible, we tried to separate text analysis discussions from the disciplinary content specific to our course. Some overlap was necessary to enmesh the two portions of the course together. But the tripartite sequence in each unit - concept, case study, practice - is intended to modularize the book enough that it could be used in other courses and contexts. This book and its contents are licensed under a Creative Commons Attribution-NonCommercial-ShareAlike 4.0 International License, meaning that you are free to share and remix any part of the work under those conditions. The book's materials are available on GitHub, where they can be copied and repurposed. Sections, especially our course-specific exercises, can be easily skipped and replaced with different tools or content. For the ambitious, you could even remix your own version that includes portions of the book and host your own GitBook course site. For more guidance in how to do so, see Adapting This Book for Another Course, but make special note of our prefatory warning about the instability of the GitBooks platform at this time.

Third, the book is an experiment in open, versioned, collaborative writing. In this, we were particularly inspired by the work of Robin DeRosa on The Open Anthology of Earlier American Literature. Our text was composed using a variety of technologies and practices relevant to digital humanities: markdown, HTML/CSS, version control, GitHub, and more. The authors had varying degrees of familiarity with these topics, and this book served as object lesson in how to generate new research and teaching materials while also developing new skillsets. The GitBook Editor, in particular, was crucial for enabling us to polish technical skills in a way that did not detract from the forward momentum of writing. The two authors are in different fields (Brandon Walsh is in English

and Sarah Horowitz is in History); accordingly, you will see vocabulary and examples that come from our different disciplinary backgrounds. But as we stress to students, although we may at times use different terms for essentially the same thing (close reading vs. primary text analysis) and have different knowledge bases, we are united by the same interest in using text analysis to explore meaning and context.

The workbook is not meant to exhaust the topic of digital text analysis: quite the contrary. If you have more than one credit of class time at your disposal, you will have much more room to navigate and explore. If your students come in with a base-level of programming knowledge, you will likely be able to skip portions of the book. The book provides only one, surface-level introduction to text analyis. There are many other approaches to the topic, and we reference some of our favorites in the "Further Resources" section of the concluding chapter. But we hope that some will build upon these materials and find the examples laid out here to be useful in developing their own courses and workshops.

# For Students

# For Students

This is a workbook about text analysis. Many of you are probably used to analyzing texts in one form or another, whether that be by carefully considering the parts of a literary text or thinking about the words in a historical document. But even though we'll be doing both those things, we are using the phrase "text analysis" in a slightly different fashion: to talk about how we can use computers to help analyze texts in new ways.

Text analysis is often understood as one of the methodologies of the **digital humanities**, alongside other activities like creating digital exhibits and online mapping. We'll talk a lot (and you'll read a lot) about digital humanities in class. Essentially, this refers to how we are using computers and new technology in the humanities. Laura Mandell offers one helpful definition of what the digital humanities are in an interview with the *LA Review of Books*:

> But the best definition of the digital humanities, I think, is bringing digital methods to bear on humanities research and then interrogating the digital humanities by humanities research.

In the same interview series, Ted Underwood describes digital humanities as "a vague interest in technology." We will keep a kindred definition in mind as we move forward: the digital humanities involves using technology to think critically *and* thinking critically about technology. We will use new tools to think about old texts and we explore the applications, perils and pitfalls of these new methods.

You may have heard of the term **big data** to describe how scholars, businesses and governments are using vast amounts of data to understand the complexities of human behavior. What you will do in this class is learn about how you can use texts in these same ways. But at the same time, we want to introduce you to some of the ways that seemingly objective "facts" and "findings" can be misleading and the product of prejudice, error and/or flawed design. Humanities students are often very good at understanding the biases and assumptions of a text. You might not necessarily be as versed in doing the same with statistical models or charts and graphs, but we hope this class will give you some experience in doing so. You will get some exposure to working with textual data and also learn about what you can contribute to these conversations.

You may be wondering: what is this thing called **the humanities**? At one level, this is just a group of disciplines or fields of study, one that often includes literature, philosophy, history, art history, and religion, and is distinct from the social sciences (politics, economics, psychology) and the natural sciences (biology, chemistry, physics). If you search the internet, you can probably find thousands of different definitions of what unites the students and scholars in these different fields. One that we particularly like is from Daniel Boyarin and Michael Nylan (in religion and history, respectively). They propose that the humanities examines:

> the different ways that human beings have chosen or been able to live their lives as human beings.

They also suggest that what unites the humanities is a common methodology:

> The primary method for the study of humans through the investigation of their cultural products is *interpretation*....I would say that the greatest difference, as far as I understand scientific method, is that for us hypotheses emerge from the data as we study and interpret, and are

> constantly being modified and corrected, while the natural sciences seem to begin with hypotheses that they test.

This class is taught by an English professor and a History professor. You'll probably notice that we have slightly different approaches and knowledge bases and occassionally use a somewhat different terminology to describe the same things. But fundamentally, we work in many of the same ways: reading and analyzing texts and thinking about meaning and cultural context. You do too, whether you realize or not, and this class aims to help you do so in different ways.

## For Students in History 211

We suggest that you read this coursebook online as opposed to downloading it as a PDF, Mobi or ePub, since some of the embedded material will only show up online. Additionally, we may make changes to the book during the course of the term, so you want to make sure you are reading the most up-to-date version of this book.

## Further Resources

- The LA Review of Books series "The Digital in the Humanities" contains interviews with many luminaries in the field and can be a good introduction for further reading as to just what this baggy field is. We especially like the interviews of Bethany Nowviskie and of Jessica Marie Johnson.

# Schedule

# Course Schedule

## Note for Students in History 211

The following is the course schedule from the beginning of the term. It's very likely that we will change it during the course of the term, but will not update the schedule here. Please consult Sakai for the official schedule.

## Description

This course examines the intersection between scandal, crime and spectacle in 19th-century France and Britain. We will discuss the nature of scandals, the connection between scandals and political change, and how scandals and ideas about crime were used to articulate new ideas about class, gender and sexuality. In addition, this class will cover the rise of new theories of criminality in the 19th century and the popular fascination with crime and violence. Crime and scandal also became interwoven into the fabric of the city as sources of urban spectacle. Lastly, we will have an opportunity to discuss how issues of crime, scandal and spectacle resonate in the 21st century. Some of the particular events and trends this class will cover include the Diamond Necklace Affair, the trial of Oscar Wilde, the Jack the Ripper murders, and the birth of detective fiction.

Through this course, students will be introduced to text analysis and data mining for the humanities. This course assumes no prior knowledge of these skills, but asks: how can newly developed technologies that allow computers to “read” large quantities of text shed light on the past? Students will work in groups throughout the course of the term to complete a digital history project that analyzes an element of the 19th century fascination with crime and scandal.

## Schedule

### Week 1

- Introductions
- Understanding Scandal
    - Ari Adut, *On Scandal*, Introduction and Chapter 1
    - Patrick Leary, "Googling the Victorians"
    - Introduction and Issues in Digital Text Analysis in this book

### Week 2

- Scandal and Monarchy, Part I
    - Sarah Maza, “The Diamond Necklace Affair Revisited: The Case of the Missing Queen”
    - Historical Essays on the Life of Marie–Antoinette of Austria
- Scandal and Monarchy, Part II
    - Tamara Hunt, “Morality and Monarchy in the Queen Caroline Affair”

    - Find two articles dating from the Queen Caroline Affair in the 19th Century British Newspapers Collection
    - Close Reading in this book

- **First Paper Due: Analysis of a Scandal**

### Week 3

- Scandal and Sexuality, Continued
    - Ari Adut, *On Scandal*, Chapter 2
    - Edward Carson's Opening Speech for the Defense of Lord Queensberry

- The Spectacle of Punishment
    - Michel Foucault, *Discipline and Punish*, selections
    - Crowdsourcing in this book

### Week 4

- Crime and the City
    - Louis Chevalier, *Working Classes, Dangerous Classes*, selections
    - Henry Mayhew, *The London Underworld*, selections

- Female Criminality
    - Lisa Downing, "Murder in the Feminine: Marie Lafarge and the Sexualization of the Nineteenth-Century Criminal Woman"
    - Cesare Lombroso, *Criminal Woman, the Prostitute and the Normal Woman*, selections
    - Digital Archives in this book

- **Second Paper Due: Analysis of a Nineteeth-Century Archive**

### Week 5

- Detection in the 19th Century
    - Simon Cole, *Suspect Identities*, Chapters 1 and 2

- The Rise of Detective Fiction
    - Michael Saler, "'Clap if You Believe in Sherlock Holmes': Mass Culture and the Re-Enchantment of Modernity, c. 1890-1940"
    - Arthur Conan Doyle, "A Scandal in Bohemia"
    - Data Cleaning in this book

### Week 6

- Violence and Entertainment, Part I
    - Rosalind Crone, *Violent Victorians*, Chapters 1 and 3
    - *The String of Pearls*, Chapters 36-39
    - Franco Moretti, "Graphs" from *Graphs, Maps, Trees*

- Violence and Entertainment, Part II
    - Rosalind Crone, *Violent Victorians*, Chapter 6
    - Find an article on a 19th century murder from the *Times* from the *Dictionary of Victorian London*
    - Cyborg Readers in this book

- **Final Group Project Proposals Due**

### Week 7

- The Spectacle of the City, Part I
  - Vanessa Schwartz, *Spectacular Realities*, Chapter 1

- The Spectacle of the City, Part II
  - Vanessa Schwartz, *Spectacular Realities*, Chapters 2 and 3
  - Reading at Scale in this book

### Week 8

- Sex and the City
  - Judith Walkowitz, “Male Vice and Feminist Virtue: Feminism and the Politics of Prostitution in Nineteenth-Century Britain”
  - W.T. Stead, “The Maiden Tribute of Modern Babylon"

- Sex and Death in the City
  - Judith Walkowitz, “Jack the Ripper and the Myth of Male Violence”
  - Find two articles on Jack the Ripper from Casebook: Jack the Ripper
  - Topic Modeling in this book

- **Annotated Bibliography Due**

### Week 9

- The Spectacle of Race, Part I
  - Clifton Crais and Pamela Scully, *Sara Baartman and the Hottentot Venus*, Introduction, Chapters 3 and 4
  - Tressie McMillan Cottom, "When Your (Brown) Body is a (White) Wonderland"

- The Spectacle of Race, Part II
  - Clifton Crais and Pamela Scully, *Sara Baartman and the Hottentot Venus*, Chapter 6
  - Cleuci de Oliveira, "Saartjie Baartman: The Original Booty Queen"
  - Pia Glenn, "You Can’t Ignore the Degradation of Saartjie Baartman to Connect Her to Kim Kardashian. You Just Can’t"
  - Danielle Bowler, "Saartjie Baartman is not ‘The Original Booty Queen’"
  - Classifiers in this book

### Week 10

- Politics, National Identity and Scandal
  - Michael Burns, *France and the Dreyfus Affair*, selections
  - Sentiment Analysis in this book

- Scandals and Contemporary Media
  - Anita Sarkeesian Interview: "'[The word “troll” feels too childish. This is abuse]"'(http://www.theguardian.com/technology/2015/aug/29/anita-sarkeesian-gamergate-interview-jessica-valenti)
  - NSA Files Decoded
  - Adam Kirsch, "Technology is Taking Over English Departments: The False Promise of the Digital Humanities"

- **Draft of Final Project Due**

## Week 11

- No class, meetings with professors about final projects
- Crime, Scandal and Politics in the Present Day
    - Julia Angwin, Jeff Larson, Surya Mattu and Lauren Kirchner, "Machine Bias: There's software used across the country to predict future criminals. And it's biased against blacks"
    - Matt Bai, "How Gary Hart's Downfall Forever Changed American Politics"
    - Conclusion in this book

## Week 12

- Class Presentations
- Wrap-Up and Class Presentations

## Exam Week

- **Final Project and Process Paper Due**

# Issues in Digital Text Analysis

# Issues in Digital Text Analysis

# Why Read with a Computer?

# Why Read with a Computer?

How are you reading this book, right now? Even though you could have printed these pages out, the odds are fairly good that you are reading it on a screen. Digital reading is a practice that most of us take part in every day (or even every waking hour of every day), and many of us probably don't think very much about the computers that facilitate the process.

Some people, of course, do: with the vast increase in digital reading over the past decade, a number of think pieces have come out describing the negative consequences of reading on a computer. Some suggest that reading on a screen before going to bed can make it difficult to relax and fall asleep. Others sigh wistfully remembering the tangible, physical nature of books that gets lost when text is translated to pixels on a screen. Some might argue that your attention span is fundamentally changed by the kind of reading you do. Or that your ownership over what you read is in danger when your books are electronic: Apple or Amazon might delete a book you have bought from your device without your permission, but it seems far less likely that Penguin will break into your home to retroactively burn a physical copy of a book they have decided you shouldn't own.

On the other hand, digital reading is often just so much more convenient. Online newspapers mean a lot less recycling. Ebooks are often cheaper than physical copies and are delivered to us in a matter of seconds -- no waiting for your books to arrive in the mail and no need for a trip to the library or bookstore!

Regardless of how you fall in these debates, it is important to recognize that a change in format necessarily changes the reading experience. We can debate the positive or negative effects of electronic reading endlessly, but we should recognize an underlying truth: you interact with a text in different ways in print than on a screen. The same is true with any medium: in a film, you are processing images and sound; in a book you are dealing with text layout and language; with recorded music you are dealing almost exclusively with sound. The technologies that carry these texts, films, and sounds to us affect our understanding of them. The scholar Marshall Mcluhan put it succinctly:

> The medium is the message.

The technologies that transmit a message -- its text in this case -- fundamentally alter the meaning and experience of the work. And we can think about the message in richer ways by studying the materials that convey them.

This is a book about reading and technology, but not quite in the same way as described above. Rather than reading *with* about technology, we are going to discuss how we might read *through* technology. It's not just that we now have access to books, newspapers and magazines online, it's also that we have access to **so much more**: all of the books on Project Gutenberg, newspaper articles from two hundred years ago, or all the blog posts that couldn't be written before the invention of the internet.

This new access to material can be overwhelming and one of the questions of this course is how can computers help us deal with information overload. And furthermore, how can we harness technology to ask new questions of texts? For instance, let's say you wanted to know the number of times Arthur Conan Doyle used the term "Watson" in *The Adventures of Sherlock Holmes* or wanted to know what the most common word was in this short-story collection. This would be a very tedious task if you just had the hard copy of the book, but it is one you can do in seconds with computer-based text analysis programs. Likewise, these same tools can help us find patterns in texts that we might not be

aware of, or allow us to collaborate with others in the reading of texts.

More specifically, we will talk a lot about the process by which we interpret texts, by which we translate ink on a page into meaning in our minds, and about how computers can tamper with and augment that process. We will touch on a number of topics and issues:

- How can computers help us understand traditional reading processes in new ways?
- How can we find new ways of reading through technology?
- How can machines facilitate new types of collaborative reading?
- How can we use computers to understand complicated categories like emotions and themes?

The implicit claim in these bullet points is that computers affect the reading process positively, but we will also give careful consideration to the wide-ranging and compelling arguments that this is not always the case.

- How does computer-assisted interpretation undermine the very point of reading?
- Do these techniques show us anything new, or are they all fancy ways to describe what we already know?
- How does reading with technology exacerbate racial, social, and economic inequalities?

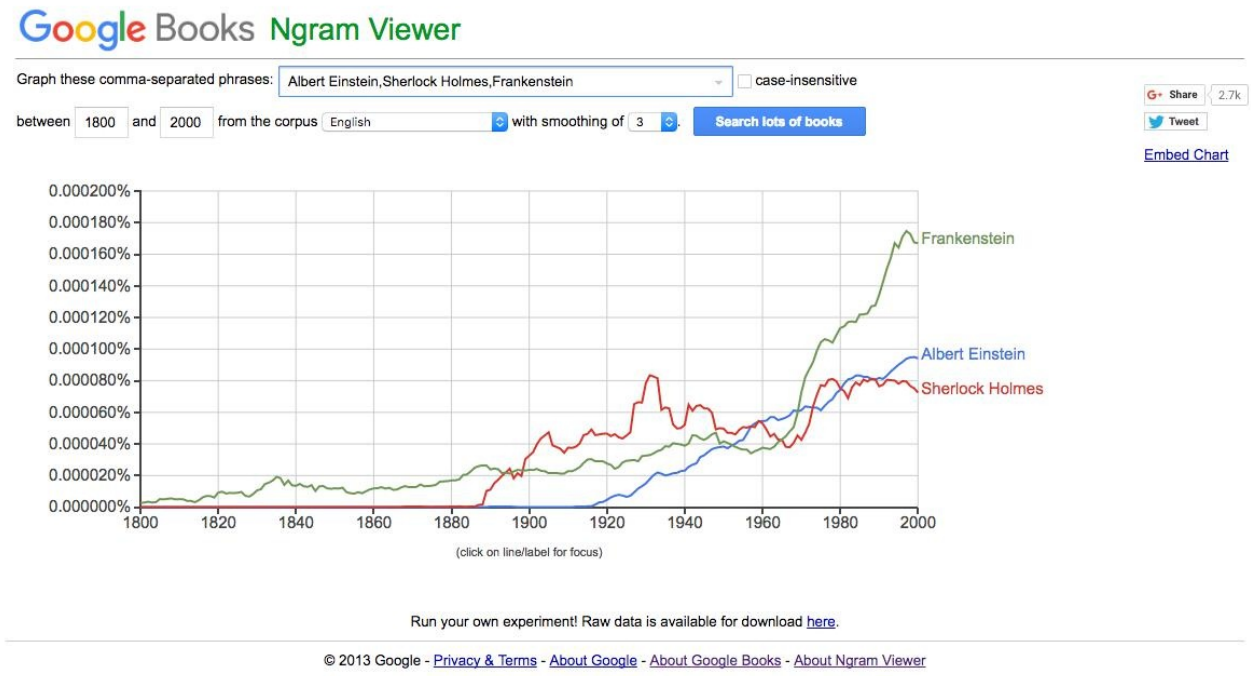
You will have to decide for yourself the answers to these questions over the course of the book.

Confused? Good. That means you're learning.

# Google NGram Viewer

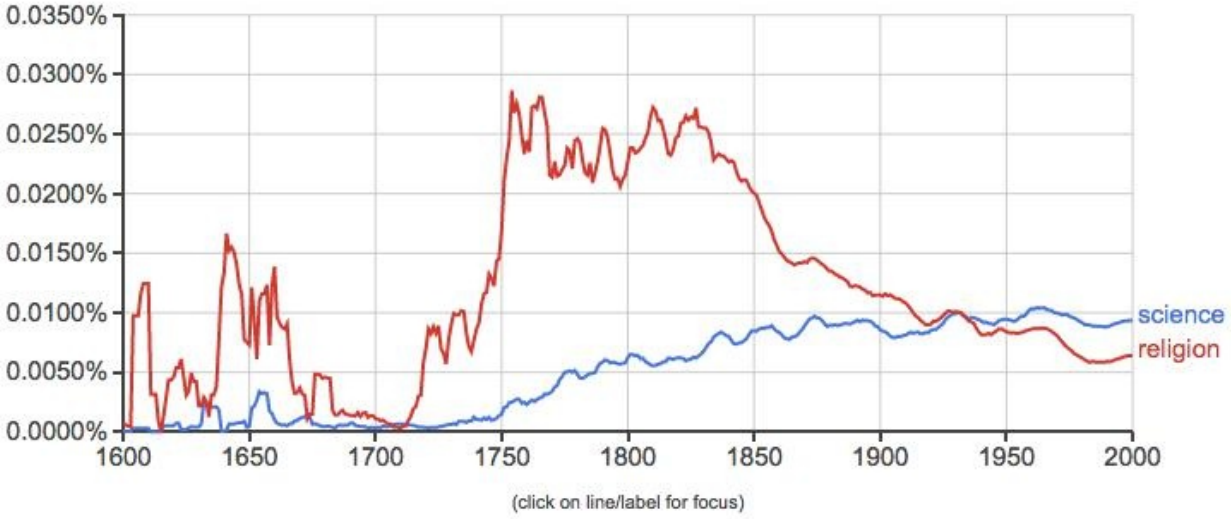
# Google NGram Viewer

The Google NGram Viewer is often the first thing brought out when people discuss large-scale textual analysis, and it serves nicely as a basic introduction into the possibilities of computer-assisted reading.




The Google NGram Viewer provides a quick and easy way to explore changes in language over the course of many years in many texts. Provide a word or comma-separated phrase, and the NGram viewer will graph how often these search terms occur over a given corpus for a given number of years. You can specify a number of years as well as a particular Google Books corpus.

The tool allows you to search hundreds of thousands of texts quickly and, by tracking a few words or phrases, draw inferences about cultural and historical shifts. If we search on 'science' and 'religion,' for example, we could draw conclusions about their relative importance at various points in last few centuries.




Looking at the graph, one could see evidence for an argument about the increasing secularization of society in the last two centuries. The steady increase of usage of the word science over the last 200 years accompanied by the precipitous decline of the word religion beginning in the mid-nineteenth century could provide concrete evidence for what might otherwise be anecdotal. But not so fast: what is actually being measured here? We need to ask questions about a number of pieces of this argument, including ones regarding:

- Corpus
- Methodology
- Interpretation

## Corpus

With any large-scale text analysis like this, the underlying data is everything. Imagine running the same word search for 'science' and 'religion' over 1000 texts used in religious schools or services. It would probably look quite different! The same would hold true if we targeted only biology, botany, and physics textbooks over the same time period. While these are fairly stark examples, the same principle holds true: the input affects the output. The data we choose for a study can skew our conclusions, and it is important for us to think carefully about their selection as a part of the process.

- What is the corpus, or set of texts, being used to generate this data?
- Where is this data coming from?

The Google NGram Viewer offers a dropdown menu where you can select a corpus to study. Our results would look a lot different depending on which corpus we selected. The corpora for these options are pulled from the Google Books scanning project (to see similar visualizations of your own corpus, you could try working with Bookworm, a related tool). This raises a number of difficulties. As Eitan Adam Pechenick, Christopher M. Danforth, and Peter Sheridan Dodds have noted, the corpus only has one copy of each book in its dataset. So things do not get scaled for circulation or popularity. A book that only sells one copy is weighted the same as a book that sells a thousand copies: they are both a single copy according to Google's methods.

The Google Books corpus has also, at times, been criticized for its heavy reliance on poor quality scans of texts to generate their data (more on this in later chapters). The computer can't infer, for example, that the mispelling 'scyience' should be lumped in with the results for 'science.' Any underlying problems in scanning or uploading texts will skew the results. In addition, the results are better after 1820. There were far fewer books published before then, and even fewer are on Google

Books.

As Ted Underwood suggests, when approached with a healthy sense of skepticism, many of these issues do not discount the use of the tool for "relative comparisons between words and periods" after 1820 or so. We can't know direct truths through the viewer, but we can still use the data for analysis. For now, just remember that graphs can appear to express fact when, in fact, the data is murky, subject for debate, or skewed.

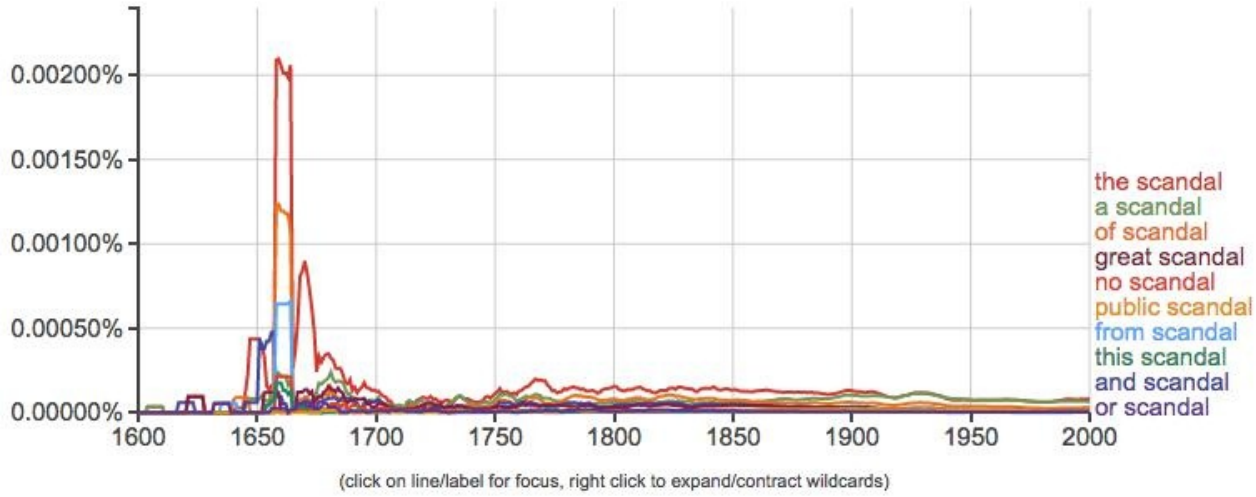
# Methodology

Even with a perfect corpus, our choices can make a big difference in the results we produce. The above search only accounts for single words, but there are more nuanced ways of using the NGram Viewer. An **n-gram** is another name for a sequence of words of length *n*. Take this short phrase:

'a test sentence.'

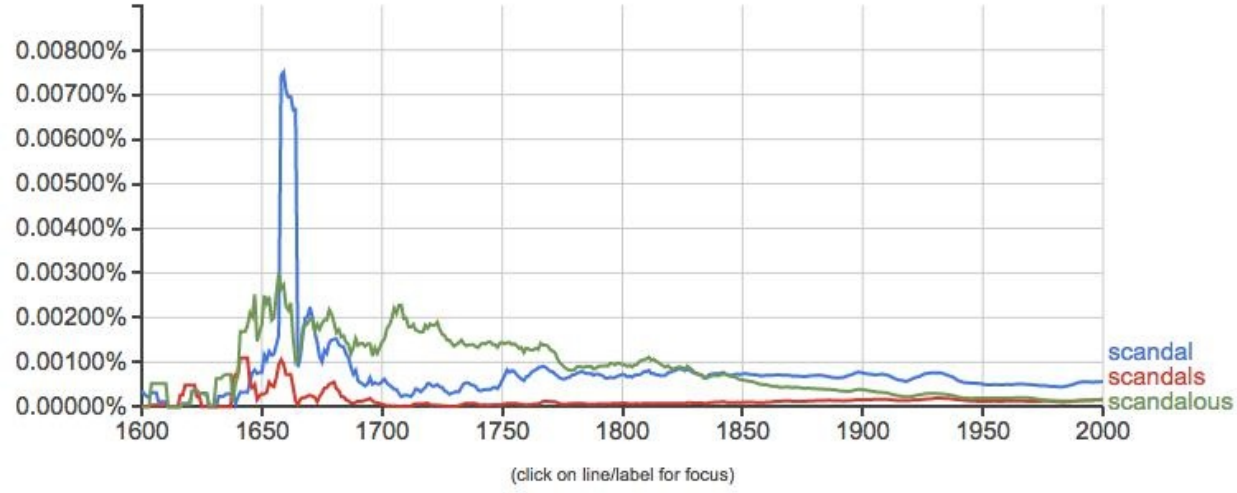
We have three n-grams of length 1 ("a", "test" and "sentence"), two n-grams of length 2 ("a test" and "test sentence"), and 1 n-gram of length 3 ("a test sentence"). Or, we could use shorthand: we have 3 **unigrams** or **tokens**, 2 **bigrams**, and 1 **trigram**. These are just fancy ways to describe different ways of chunking up a piece of text so that we can work with it. And we can do the same thing in the NGram Viewer. Take this NGram for the token 'scandal' in an English corpus:


It appears like something fairly dramatic happened around 1660 that caused a massive spike in the usage of 'scandal.' This in itself could be significant, but we might be interested in more nuanced readings of this data. We might want to see, say, bigrams containing scandal like 'political scandal' and 'religious scandal' to observe when certain types of scandals come into prominence. The NGram Viewer allows for a number of nuanced searches that you can read about here. For now, let's try out a wildcard search - '* scandal':




The asterisk in searches like this matches anything, so it will return all two-word phrases containing 'scandal' as a second word. And, handy for us, it will show us the top ten uses. In this case, they're almost all articles or prepositions: 'the scandal,' 'a scandal,' 'of scandal,' etc. And they all seem to spike around 1660 as well. We would need more information about this time period to tell exactly what is going on here, and to do so we might want to specifically exclude these common usages. Given the relative unreliability of N-Grams before 1820, this dramatic uptick might be due to just a few works that used the term "scandal" around this time -- and might not be representative of larger patterns.

We might also want to look at different forms of the same word. After all, the above search only captures the singular form of 'scandal', but any word can occur in multiple forms over the course of a corpus. The NGram Viewer can account for this as well:




That massive spike we see in the use of 'scandal' is not quite matched by other forms of the word. In particular, the adjective form 'scandalous' enjoys more usage until the mid-nineteenth century. Maybe scandal as a noun, as an idea, as a thing unto itself explodes onto the scene in the mid-nineteenth century, wereas before it was something more a thing attached to other people, places, and events.

To drill down more deeply into another term relevant to this course, check out this ngram of the word 'crime' in the English corpus:


According to this chart, after a drop during the early-eighteenth century, English writers discussed crime more consistently and ubiquitously than ever before. But what about authors writing in other languages? Here is the same search in French.

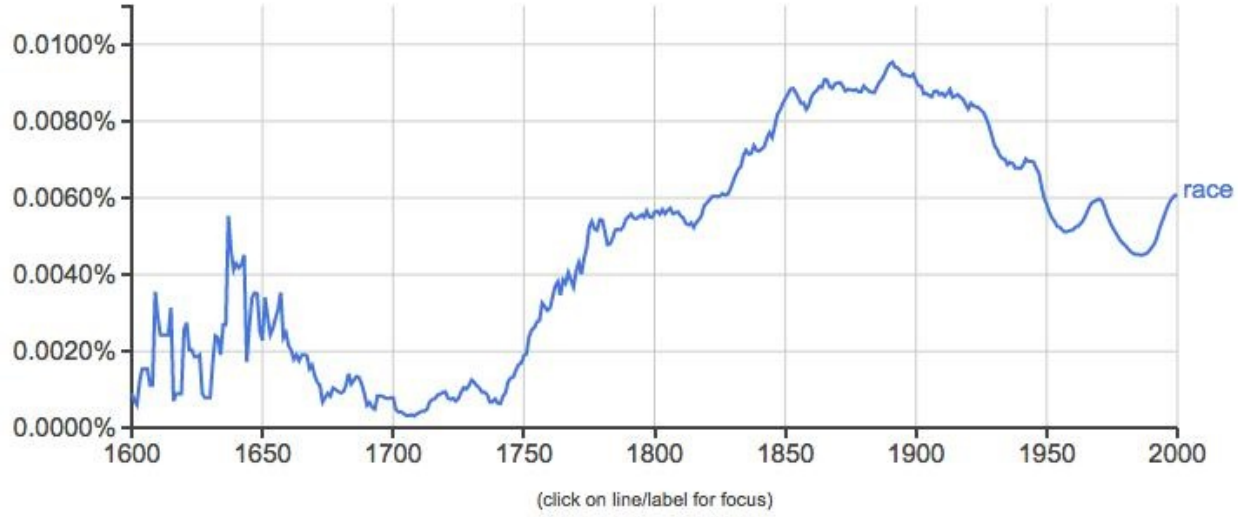

The general trend of more mentions of crime in the 19th century than the 20th holds true in both the French and English corpora. However, if you pay careful attention to the y-axis you will note that French authors actually are mentioning crime far more frequently relative to the rest of the writing at the time. The trends are similar, but the percentage of times 'crime' shows up is much higher in France. In England during this time, uses of the word hover around 0.0045. French writing mentioning 'crime' is over double that percentage during the same period, and it does not dip down to that number until 1880.

You will also note a different trajectory to these two N-Grams. In the English-language corpus, mentions of "crime" go down gradually over the course of the nineteenth century. In contrast, there is a big spike in the French corpus which starts going down quite dramatically in the 1830s. When you read portions of Louis Chevalier's *Laboring Classes and Dangerous Classes in Paris during the First Half of the Nineteenth Century* later in the term, you'll get a sense of why this interest in crime surges in the early nineteenth century and then dies down.

If we were using N-Grams for more than just a demonstration, we would want to do a lot more research and thinking about both language and history. For instance, in the above example, is the fact that French authors seem to be using "crime" more often than English-language ones due to a difference in language and usage? Perhaps English authors often use a synonym for crime, whereas

French ones do not? Or does it reflect the fact that French authors were more concerned with crime than English ones?

Or, let's say we were interested in history of scientific racism in European and American thought. In that case, we might want to know about the trajectory of the word "race" over time.




This N-Gram shows an increasing use of this term over the course of the eighteenth and nineteenth century, peaking around 1890 and then gradually declining in the twentieth century, albeit with some upswings. On the one hand, scientific racism had one of its heydays in the late nineteenth century, so maybe this N-Gram shows this historical trend. But as soon as you think more about this topic, you would realize that it's a lot more complicated that this. For one, it might strike you as a little odd that the line dips in the late 1950s and early 1960s, an era when the Civil Rights movement was emerging. Were people really writing *less* about race then than before? Alternatively, the term "race" can mean a lot of different things; in this case, the results we are interested in (the categorization of people according to their phenotypes) are undoubtedly getting jumbled in with references to sporting events and elections. And as soon as we started doing research on the history of scientific racism, we would learn that writers used the term "race" to refer to groups of people in different ways in the eighteenth century than they did at the end of the nineteenth century. We would also want to think about terms that are associated with or used as synonyms for race. Maybe authors in the twentieth century were using other words to talk about race? If so, what might those be?

So, language changes over time. A single word might radically change in usage over the centuries in ways that skew our results. We also use different terms over time to describe the same phenomena. These are all things we would want to say a lot more in any interpretations using N-Grams. We would also need to consider what they can (and cannot) tell us and think about potential problems in my reading. But hopefully the implications of the technology will be exciting to you nonetheless.

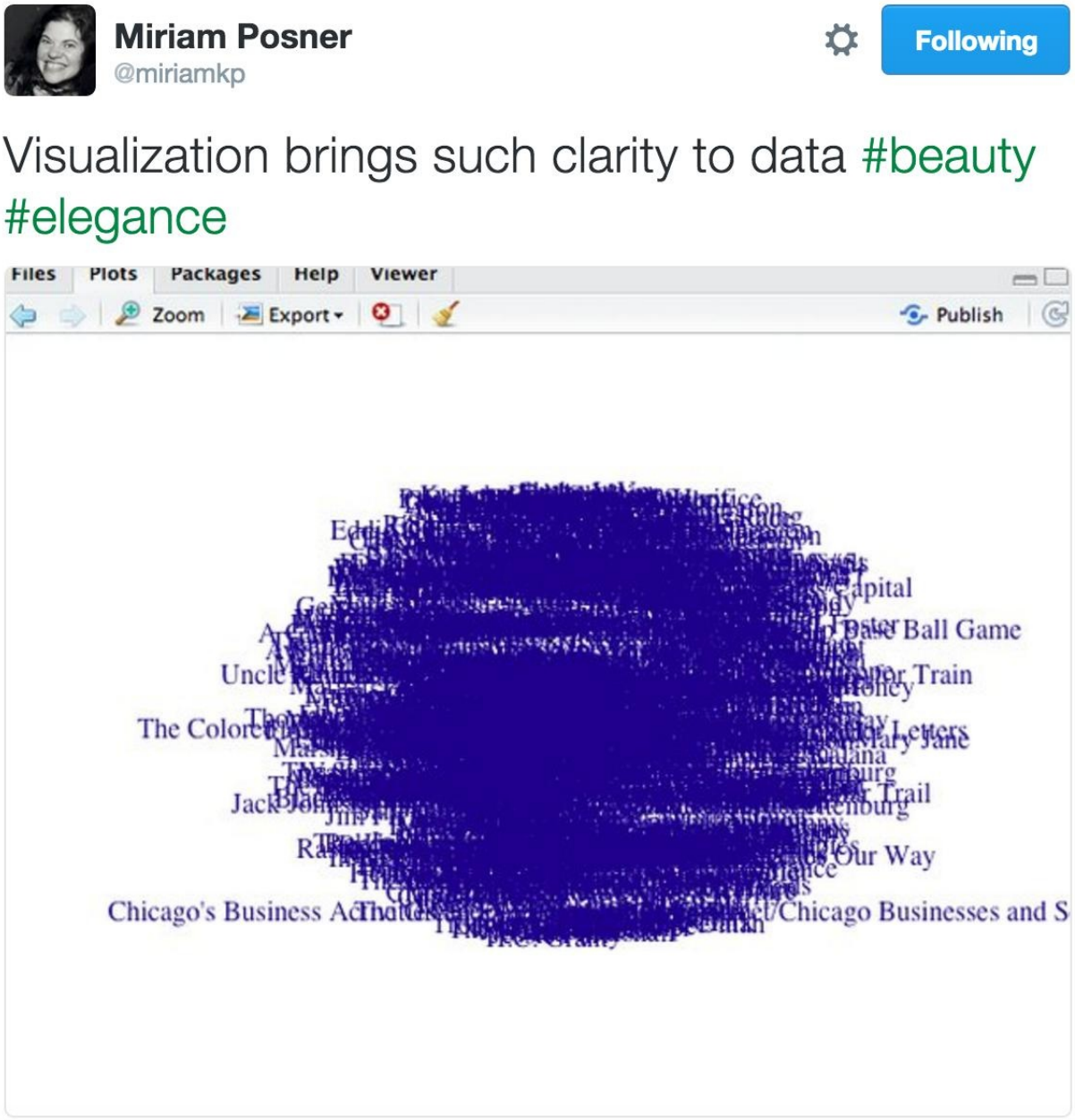
- Digital methods can allow us to make observations about vast numbers of texts. Far more than you would be able to read yourself.

That last phrase should cause some alarm: we haven't actually read any of these texts, but we are making observations about them nonetheless. We hope you will think deeply about the implications about such an act.

- What does this form of reading lose?
- What does it gain?
- How it can it be approached in ways that minimize the former and maximize the latter?

## Interpretation

Of course, these graphs mean nothing on their own. It is our job to look at the results and describe them in meaningful ways. But be critical of what you see. You might find something interesting, but you might be looking at nonsense. It is your job to tell the difference. Beware of **apophenia**, the all to human urge to look at random data and find meaningful patterns in it. You can find wild patterns in anything if you look hard enough. After all, visualizations can confuse as much as clarify. Numbers and graphs do not carry objective meaning. Miriam Posner summarized it pithily on Twitter once:




Always think. Never let a graph think for you.

## Further Resources

- Ted Underwood on "How not to do things with words" for helpful criticisms of studies employing the Ngram Viewer.
- Danny Sulivan on "When OCR Goes Bad: Google's Ngram Viewer & The F-Word" for more on Google NGram and OCR.
- Geoff Nunberg on "Google Books: A Metadata Train Wreck" for more problems with the viewer.

# Exercises

# Exercises

1. What is your own background with computers? Interpret this question as broadly as you'd like.
2. Take a few minutes to reflect on how you read and what happens when you are reading. Then describe your process in 3-5 sentences.
3. What kinds of technologies do you use for reading? Do you feel that your use of different technologies and different ways of reading (reading a physical book versus reading something on your computer versus reading it on an e-reader) changes your experience of reading?
4. How do you imagine that reading was different in the nineteenth century than it is today?
5. Use the Google NGram Viewer to track two different terms that interest you. Interpret the results. What seems interesting? What kind of historical events might account for any shifts that you see? What terms rise or fall and when? Feel free to search on Wikipedia or Google to back up your interpretations, but don't overdo it. Keep your responses to 3 sentences per term. Provide a screenshot for the NGram search you carried out.

# Close Reading

# Close Reading

# Close Reading and Sources

# Close Reading and Sources

Text analysis is something that we all engage in, whether we realize it or not. The term is broad and capacious and encapsulates a variety of different activities. Even something as simple as slowing down when you see a stop sign is a kind of text analysis: doing so means you have parsed the meaning of the words on the sign and reacted accordingly.

Indeed any of the following, related activities are forms of text analysis:

- Paraphrasing a text
- Searching for hidden meanings in a text
- Adapting a text and reflecting on it
- Examining the details in a text

This last point is worth pausing over: **close reading**, in particular, is often proclaimed as one of the primary analytical tool of scholars and students in the humanities. To read closely means to give careful attention to the various components that make up a text, ones which cause us to think or feel a certain way about it. Close reading relies on a core principle about the text under study:

- Everything about the text matters, whether the author intended for it to matter or not.

Consider the following thought experiment. One day you come home to find the following note from your roommate on the counter:

> took care of these dishes? Thanks.

Next to the note: dirty dishes. Was your roommate in a hurry and actually asking you to wash dishes? Or were they sarcastically trying to give you grief for not having done your part? Lots of questions. To imagine responses to them you might employ a range of assumptions and interpretations depending on the scenario:

**Context**: you have been growing more and more irritated with your roommate for some time now. Their actions just really get under your skin: dirty dishes, laundry everywhere, the works. They clearly meant the note as an insult.

**Author**: your roommate is actually a great person and would never leave a passive aggressive note. In fact, they probably meant it as a joke.

**Text**: Take a look at that question mark. And then the curt second sentence. Your roommate put those things there on purpose to be rude.

The list could go on and on. We employ a similar range of skills when we read anything, be it fiction, poetry, or historical documents. Close reading might be best described as an activity in which a reader simply lets no detail of the text go unquestioned. The best way at approaching a good close reading is by asking (and attempting to answer) questions about every piece of a text.

Take a sentence from the 1775 *Anecdotes on the Countess du Barry*, a *libelle* (which you can find here) similar to the ones discussed in Sarah Maza's “The Diamond Necklace Affair Revisited: The Case of the Missing Queen.” Mme du Barry was a prostitute who was Louis XV's mistress at the end

of his reign (1715-1774). Here is how the Count du Barry tells one of Louis XV's courtiers that he has a woman in mind for the king:

> "I've got your business for you. You know I don't lack taste. Trust me: you come to dinner at my house and tell me that I'm a cad if I don't give you the most beautiful woman, the most fresh, the most seductive; a true morsel for a king."

In beginning a close reading here, I might ask:

- What adjectives and nouns describe Mme du Barry here?
- More specifically, what does it mean that she is compared to a "business" or a "morsel"?
- If she is a piece of food, what does that mean about the relationship she might have with Louis XV?
- Why is she not named here?
- If you read the rest of the text, you'll see that most of the language in this excerpt is flowery -- but not the Count du Barry's words. What does that suggest about who he is and what his character is like?

You can answer these questions any number of ways, and this ambiguity is part of the point. Close reading as a method is a way of training yourself to look for details, the evidence that you will use to interpret a passage, but how you use them depends on you. This evidence becomes the material you use to produce an analysis of your own (sometimes also called a close reading). Using the questions about *Anecdotes on the Countess du Barry*, I might make the argument that these sentences establish her as an object, a commerical good or a commodity for the king's consumption. I might also think that the Count du Barry's words render him as vulgar and coarse, a figure unworthy of contact with the court of Versailles.

## Primary and Secondary Texts for Historical Analysis

In addition to reading texts closely, you also want to think about the kind of text you are working with and its relationship to its historical context. For starters, you need to know if the work you are reading is a **primary** text or a **secondary** text. The Healey Library has a good set of definitions:

> **Primary Sources** are immediate, first-hand accounts of a topic, from people who had a direct connection with it. **Secondary Sources** are one step removed from primary sources, though they often quote or otherwise use primary sources. They can cover the same topic, but add a layer of interpretation and analysis.

Sarah Maza's article is a secondary text, whereas the *Anedcotes*, discussed above, is a primary text.

Reading primary texts is absolutely invaluable, particularly in the study of history. There is no better way to understand events in the past than by examining the sources – whether journals, newspaper articles, letters, court case records, novels, artworks, music or autobiographies – that people from that period left behind. However, you need to approach primary sources with care and as something other than a 100% accurate representation of the truth. For instance, in reading the *Anecdotes*, you might ask: did the author actually witness the events he or she was describing? Probably not. In that is the case, what can this document help us understand? And what can't we use it to do?

Thus, you want to read primary sources with a series of questions in mind. The following is adapted from a guide provided by Carleton College:

1. What implicit and explicit messages does this text contain? What did the author choose NOT to talk about?

2. What do you know about the author? How might his or her beliefs or background have affected the writing of and views in this document?

3. Who constituted the intended audience? Was this source meant for one person's eyes or for the public? How does that affect the nature of the source?

4. Is it prescriptive (telling you what people thought should happen) or descriptive (telling you what people thought did happen)?

5. Does it tell you about the beliefs\/actions of the elite, or of “ordinary” people? From whose perspective?

6. What historical questions can you answer using this source? What questions can this source NOT help you answer? What are the limitations of this type of source? What historical perspectives are left out of this source?

7. What assumptions do you see being made? What surprises you about this text?

For instance, take the following passage from the first paragraph of the *Anecdotes*:

> Advancing age and the ability of a great prince to satisfy all his passions had dulled his attraction towards women. But this need, though diminished, continued ... The doctors assured the King that it was dangerous to give up so abruptly a pleasure necessary for his existence.

At one level, this work is giving an account of how Louis XV and the Countess du Barry began their liaison. But these sentences also have an implicit message: the king's sexual desire for women was natural and indeed necessary to his well-being. This view that the king needed to have a mistress for the sake of his health may be surprising to you and it certainly reveals a set of assumptions about extra-marital activity at the time. So if we can't take this primary source as an accurate representation of the relationship between du Barry and the king, it does serve as a fascinating window into into the culture of late eighteenth-century France.

## Digital Reading

Interrogating sources in this fashion is just one mode of understanding a text. It relies on the idea that sustained attention to a document will allow you to understand new things about it. This same approach can be applied to virtually any object of study, be they books, films, music, or ideas themselves. Our primary motivation in this book, then, is how the process can be changed with the introduction of technology. You might start by asking a series of questions about how close reading might interact with digital tools.

- Can we still read closely if the computer is doing the reading for us?
- How might the computer change the kinds of close readings available to us?
- Can we close read new things using digital tools that we couldn't before?

# Prism Part One

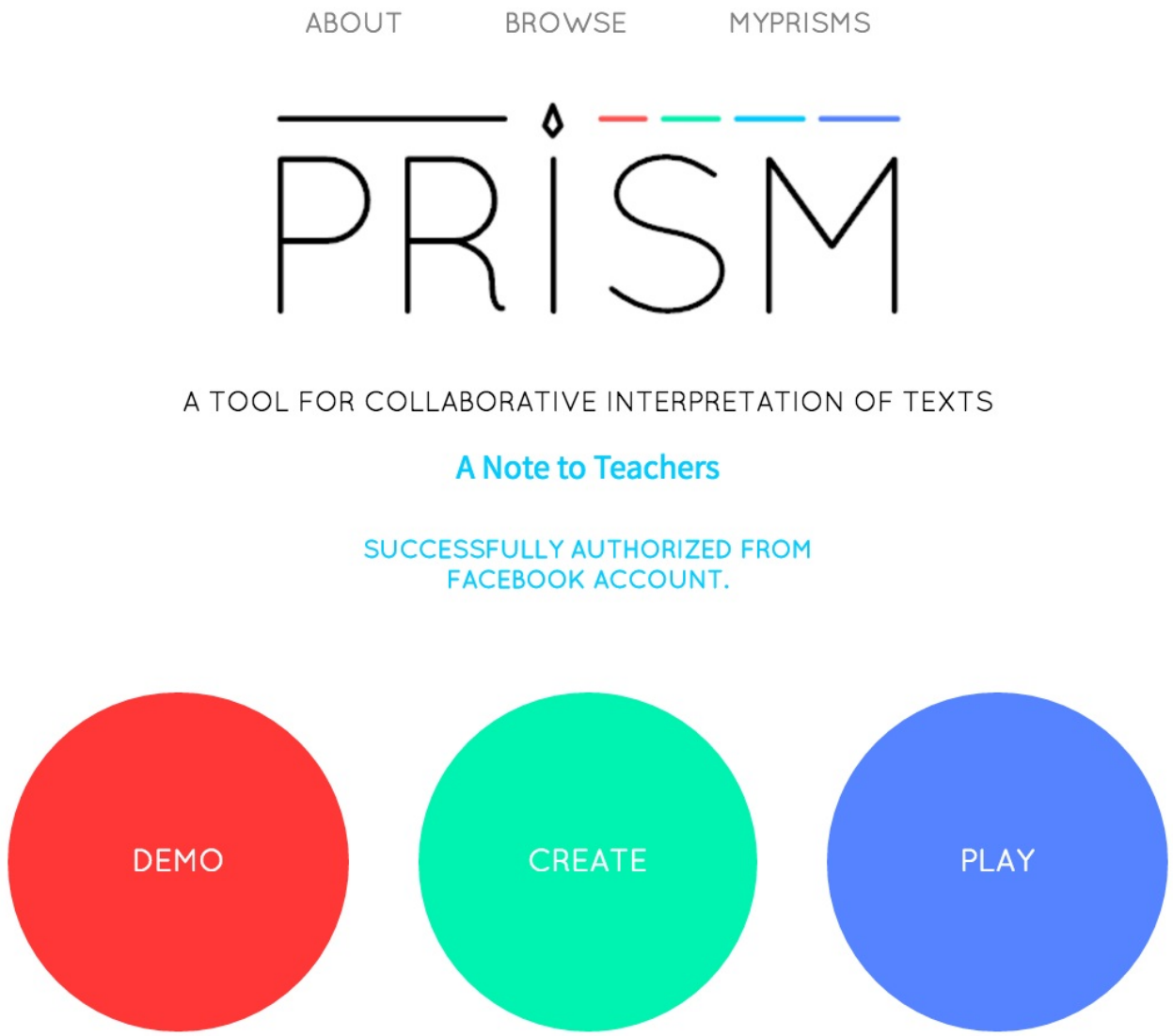
# Prism Part One

Prism is a digital tool that enables readers to think about how they interpret texts in new ways. The project grew out of a series of conversations and exercises carried out by Johanna Drucker, Bethany Nowviskie, and Jerome McGann at the University of Virginia. Every member of the group would receive a copy of a single text as well as a transparency and a few highlighters. One person would give the highlighters to the group and say something to the effect of, "OK. Read this passage, and mark passages that seem to suggest 'democracy' with the green highlighter and 'anarchy' with the blue." With the transparency laid over the passage, the readers would all mark their own copy as they read. The marking process would end at a certain point, and the transparencies would be collected.

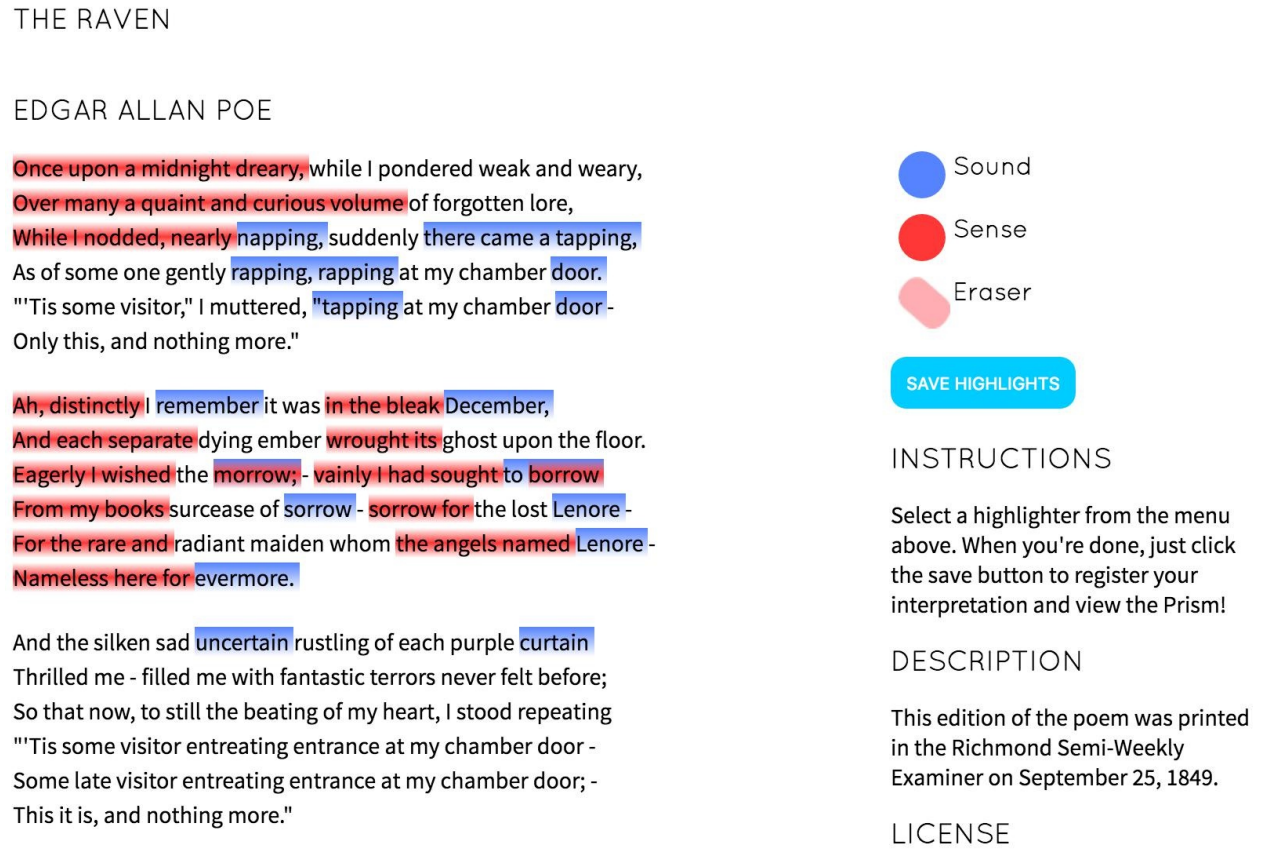
The transparency game crystallizes a very basic element of textual analysis:

- When we close read, certain textual components - phrases, words, characters - make us interpret a document in a particular way.

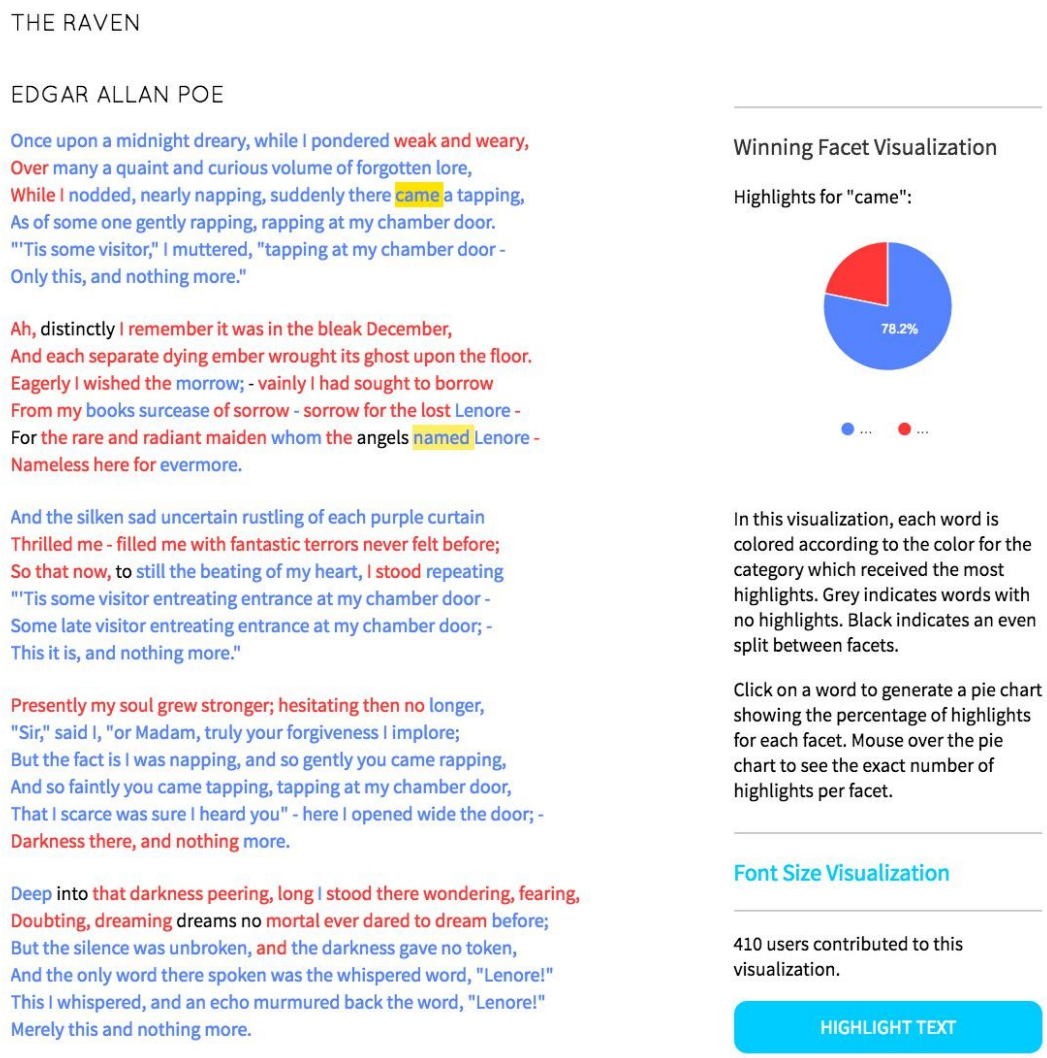
The game asks you to make graphic representations of these decisions, to identify the words that make you think or feel a certain way. By lifting the transparency off the text, you are left with a series of colors that correspond to your reading of the document. You have traced the broad outlines of your interpretation onto the page.




Prism is a digital version of the same game. Given a choice between a few predetermined categories, Prism asks you to highlight a given text. In this Prism example, readers are asked to mark an excerpt from Edgar Allan Poe's "The Raven." By selecting one of the buttons next to the categories on the right, your cursor will change into a colored highlighter. Clicking and dragging across the text will highlight it in the same way that you might if you were reading a print version.

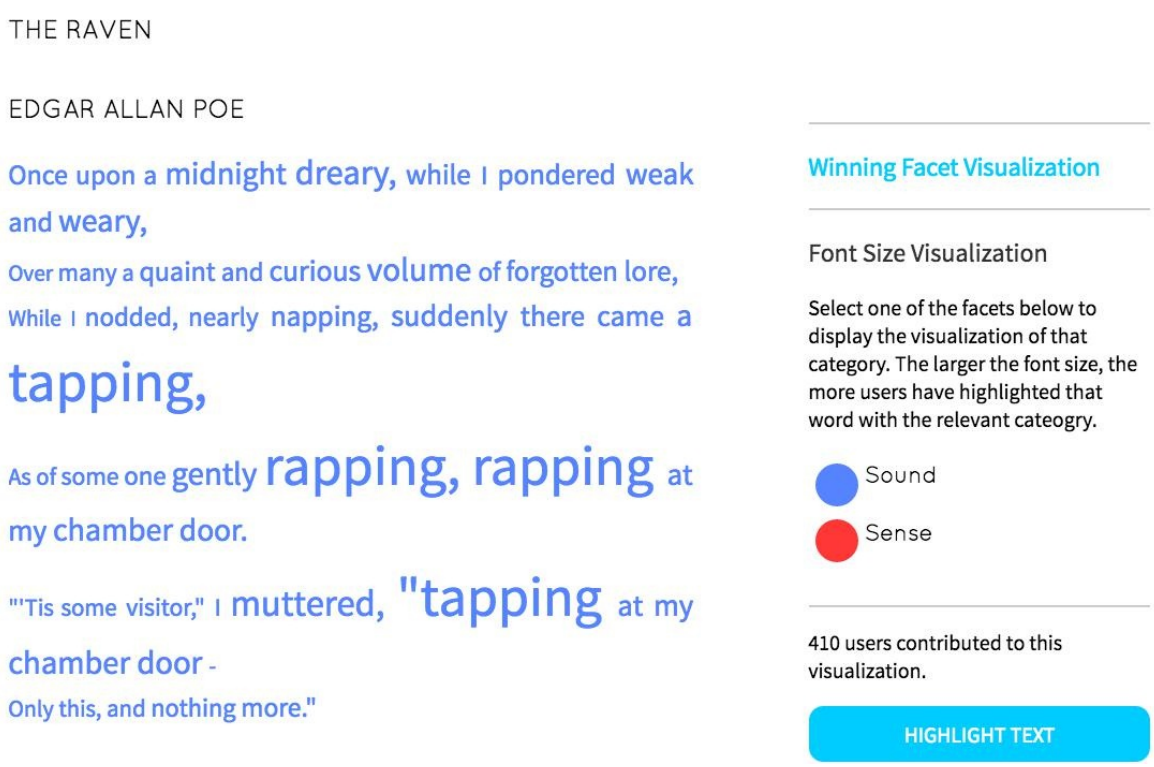



After you click "Save Highlights", the tool combines your markings with those everyone else who has ever read the same Prism text made to help you visualize how people are marking things. By default, Prism will bring up the **winning facet visualization**, which colors the text according to the category that was most frequently marked for each individual word. Clicking on an individual word will color the pie chart and tell you exactly what percentage the word got from each category.




Seeing a graphic representation of the reading process might help you to notice things that you might not otherwise. For example, here you might notice that people tended to mark passages containing first person pronouns as "sense." Is it because "sense" implies thinking? Phrases like "I remember," "my soul grew," and "I stood there wondering" do suggest an emphasis on introspection, at the very least. Did you mark the same phrases, or did you select other passages?

Prism comes with two visualizations baked into it. To change visualizations, click the "Font Size Visualization" button on the right sidebar. The **font size visualization** lets you see which parts of the text were more frequently thought of as belonging to a particular category: Prism resizes the text to reflect concentrations of reading. So in this example, where readers were marking for "sound," they tended to mark rhyming words more frequently.




Makes sense, and you might have done the same. By selecting the other category, you could check out what readers tended to mark for "sense."

By design, Prism forces you to think more deeply about the categories that you are given for highlighting. The creator of this Prism wants you to mark for "sound" and "sense" - categories that relate to Alexander Pope's famous formulation of poetry from *An Essay on Criticism*. In it, Pope suggests that the sound of a poem should complement the underlying meaning of the poem. So the creator of this game wants you to try and pinpoint where these categories overlap and where they depart. You might not have known this context, though you might have intuited elements of it. Guided reading in this way might change how you would otherwise read the passage, and the absence of clear guidelines complicates your experience of the text.

- How would reading be different if you do not know the exact meanings behind the categories?

# Exercises

# Exercises

Practice your close reading skills on the following short passage:

---

Excerpt from "The Respectful Petition of an Humble Subject to her Majesty Queen Caroline," The Morning Post, September 09, 1820, issue 15439.

That the prosperity of a nation does, very materially, depend on the preservation of the moral virtues has ever been indisputable, and it would be totally unbecoming your MAJESTY's sex to question that morality principally depends on females. Now, with all due deference for your MAJESTY's enlarged sentiments and highly cultivated understanding, is it fit, or proper, that whilst accusations of the grossest nature, alike reflecting on your MAJESTY's virtue and delicacy, and solemnly discussing in the highest tribunal of the empire — whilst wives blush to hear their husbands read the tale — whilst mothers hide the details of profligate prostitution from their families — whilst virgin innocence blushes at the mention of England's QUEEN - whilst the eye of modesty is averted, and chastity throws her mantle over the display of imputed, boundless passion, is it befitting a woman - can your MAJESTY think at all, and reconciling reflection with regal dignity and female importance, pronounce it blameless, that bearing the weight of these heavy charges, your MAJESTY should parade the streets of this metropolis, triumphantly proving your possessing a front which no respect for yourself, or consideration for the guiltless, can induce you to conceal? ... Oh! Madam, there are females in our island who earn their daily bread by daily toil, that would not exchange conditions with you, whether viewing you in a Neapolitan theatre, rioting in the mirthful buffoonery of your villa at Como, or drawn by half a dozen richly caparisoned studs... Though late, Madam, still deign to take counsel, and be persuaded, that the vulgar shouts of a shameless mob are not the hymns of a sensible, reflecting populace ; nor deem the ditties of itinerant ballad-mongers from the purlieus of St. Giles's the carols of those who esteem retirement from public gaze, and absence from the page of notoriety, a woman's most amiable sphere.

---

- Write a paragraph describing your reading process in as great detail as you can manage. What kinds of things do you notice about the passage? Are you looking for particular things? What goes through your head?

- Then read the passage two more times and repeat the same exercise, writing down what goes through your head in detail.

Then go to our class Prism. Highlight the Prism according to the rules of the game and then respond to the following questions:

- How did you read differently when using Prism? Can you imagine other variations of the tool? How might things with five interpretive categories, for example?

- Say something interesting about the results that Prism gives you. What new insights does it give you into the text?

- Think more critically about the categories we have chosen for you to highlight. What assumptions do you think we have about them? How does pairing these two categories change how you think about the categories themselves?

# Crowdsourcing

# Crowdsourcing

# Crowdsourcing

# Crowdsourcing

Think of a common scenario: you are buying something online and, before the system will allow you to check out, you have to enter some text, transcribe some audio, or otherwise prove that you are not a robot. Doing so only takes a few seconds of your time, but such transactions happen millions of times everyday on the internet. The combined energy wasted on such simple interactions is astounding.

Now imagine that you could take all those hours of human labor and put them to something useful. reCAPTCHA aims to do just that, and you've probably unwittingly used it. Those human-validation tasks we described in the last paragraph? The chances are pretty high that, when you carried one out, you may have unwittingly corrected an image transcription, helped provide a test set for artificial intelligence, or helped to make Google Maps more precise. The next time you fill out a text box to prove that you are a human, you might take a closer look at what you are doing and ask, "what is my work being used for?"

**Crowdsourcing**, broadly defined, can be thought of as the application of many different people to a single problem by having them each work on a piece of the project. The most common type of crowdsourcing is used to correct text transcriptions. When you scan an image with text in it, sophisticated computer programs run over the image to make their own best guess as to what that text might be based on vast quantities of information. This process is known as **optical character recognition (OCR)**, and these guesses are often incorrect: given many variations in font, ink type, contrast, and more, the task is actually very complicated and difficult. These mistakes are often called **dirty OCR**, and an example might look something like this:

"Qi-jtb"

That might not mean a lot out of context, but alongside an image you could probably piece together the word it was meant to represent from the original print artifact. Ryan Cordell fixes on this particular poor scanning of the first word in the famous phrase "Quoth the Raven" from Edgar Allen Poe's "The Raven" as an example of the problems that scanning can present for studying historical documents. Such errors complicate a lot of digital text analysis: a search through the document for "Quoth" would not return this instance unless someone (or something) cleaned it.

You will learn more about this and other problems with digital data in our chapter on "Data Cleaning". For now, suffice it to say that correcting such errors is long, tedious work. Doing so doesn't require much intellectual energy for a human, but it does take a lot of time. On the other hand, these correcting tasks would be very difficult for a computer to do accurately, but computers can manage the large scale of the work with little trouble. Projects like these use a technique called **microtasking**, meaning they direct the human energy of many many people to finite tasks. Microtasking projects find solutions to big problems by making them into small tasks that individuals can solve. OCR Correction is a common scenario for such projects: Transcribe Bentham asks users to prepare corrected versions of the papers of Jeremy Bentham's unpublished manuscripts, and 18thConnect has developed Typewright to correct a vast number of early modern materials. Each of these projects relies on individual people correcting texts one piece at a time.

Whereas microtasking projects ask users to work on a problem already laid out for them, **macrotasking** projects are lead by the interests and aims of the group itself. *Wikipedia* is probably the most famous example of crowdsourcing, and it falls under this category. Its many users apply their

energy to common problems: the production of knowledge regarding particular topics. But *Wikipedia* is different from other forms of crowdsourcing in that it has no clear goal in sight. We can never write down all of human knowledge: instead, *Wikipedia*'s devoted users will continually work to develop a better understanding until the website goes offline. The user community creates its own goals, priorities, and tasks, all of which can lead to systemic problems: the articles online do not necessarily reflect the inherent interest of the material but, instead, the interests of the community of editors. (In the case of Wikipedia, this means that it has a significant gender problem.) Whereas microtasking projects are about really small, repeatable problems, macrotasking problems are fundamentally different in kind.

It is worth pausing over all of these examples to consider the labor going into them. We are talking about an incredible amount of energy and work that is essentially volunteer. If we go onto *Typewright* and help transcribe an eighteenth-century text, that is time that we could have spent doing something else, including work for which we could have been compensated in more explicit ways.

- Is it enough that the users are contributing to the public good for these projects?
- At what point does volunteer labor become exploitation?

In many cases, these digital projects cost vast sums of money, and, so the critique goes, these funds could have provided for actual paid employees instead of volunteers. Some of these crowdsourcing participants may not even have realized they were working. In the case of *Recaptcha*, you probably unwittingly volunteered your time for a crowdsourcing project without even realizing it.

- What are ethical practices for conducting volunteer projects on such a scale?
- What would it take for you to feel adequately compensated for your time?

These are open questions with no clear answers, but they are worth keeping in mind. We think this awareness and self-reflection must be the foundation of ethical ways of engaging with projects like these. After all, *Typewright*, *Recaptcha*, and *Transcribe Bentham* produce great results, but they do so by employing human energy to fairly menial tasks. *Wikipedia* raises other important questions about crowdsourcing:

- Can the crowd do more?
- What happens when we give control of the project over to the crowd?

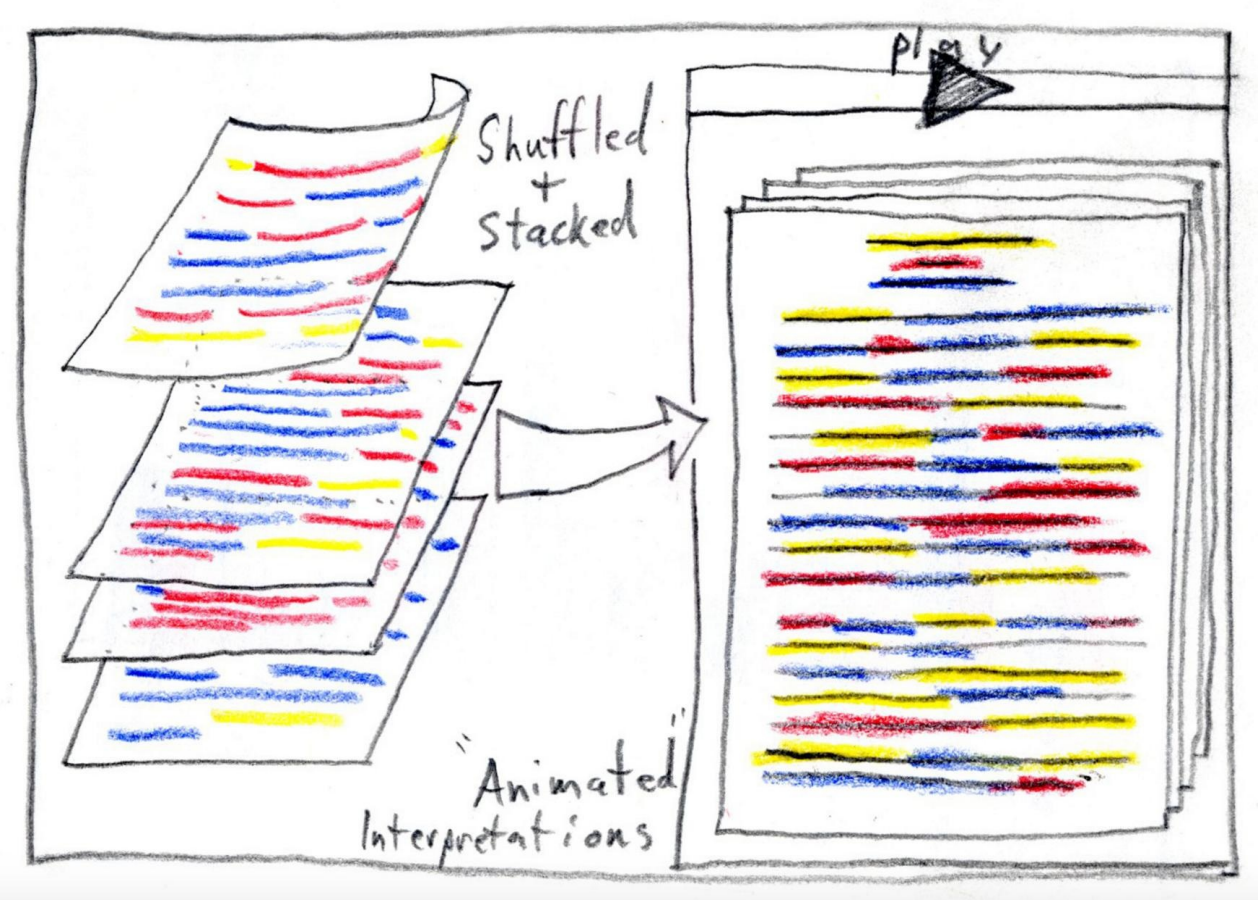
## Further Resources

- Brandon Walsh, et al. have a piece on Prism and crowdsourcing with a useful bibliography for pointing to other crowdsourcing projects.
- Mia Ridge's book on *Crowdsourcing our Cultural Heritage* is fantastic on crowdsouring in a cultural context.

# Prism Part Two

# Prism Part Two

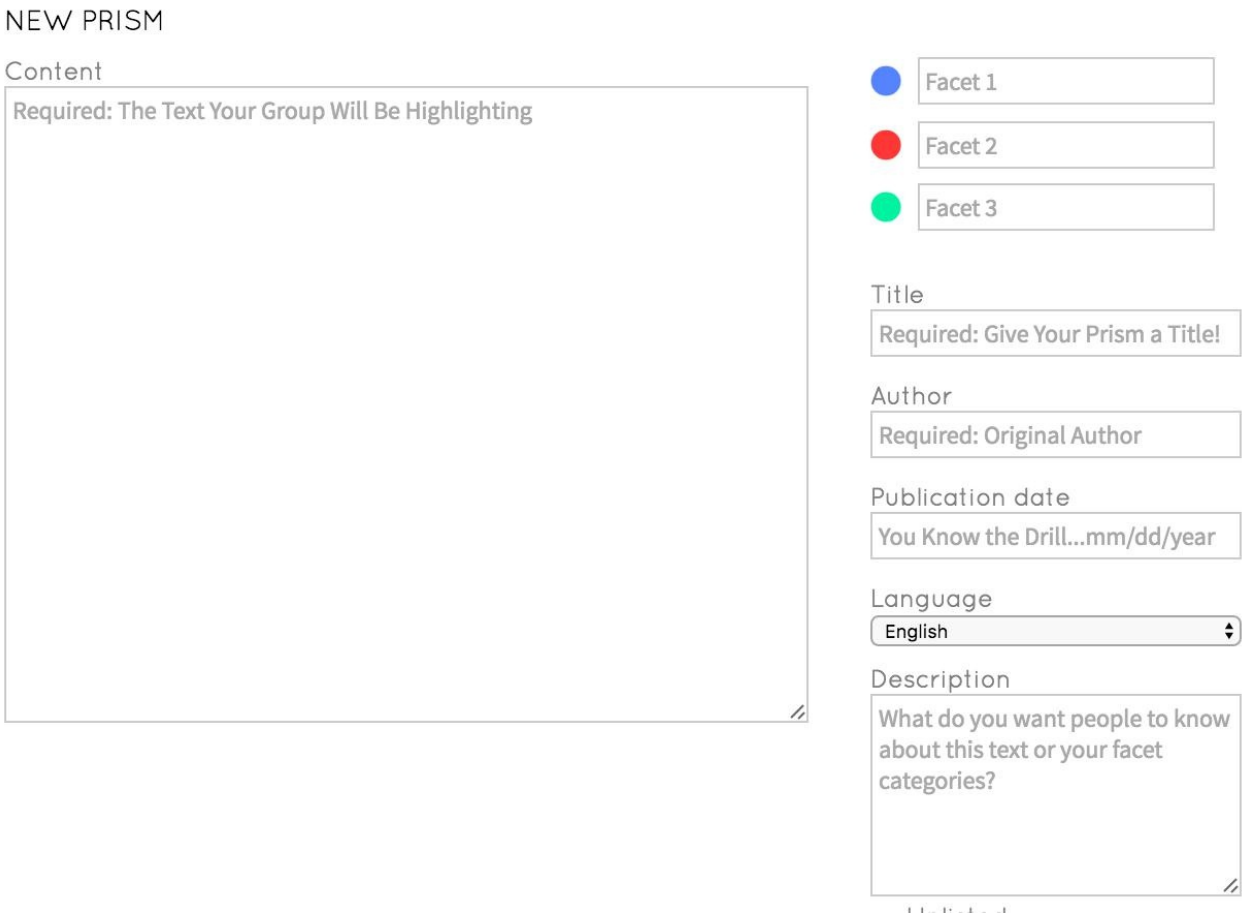
Think back to Prism and the transparency game. So far we have only really focused on the single transparencies and the interpretations one individual supplied. But the crucial last element of the game involves collecting the transparencies and stacking them. Hold the stack up the light, and you get a whole rainbow: you can see what everyone thinks about the text. Prism's visualizations offer one way of adapting this activity to a digital environment.



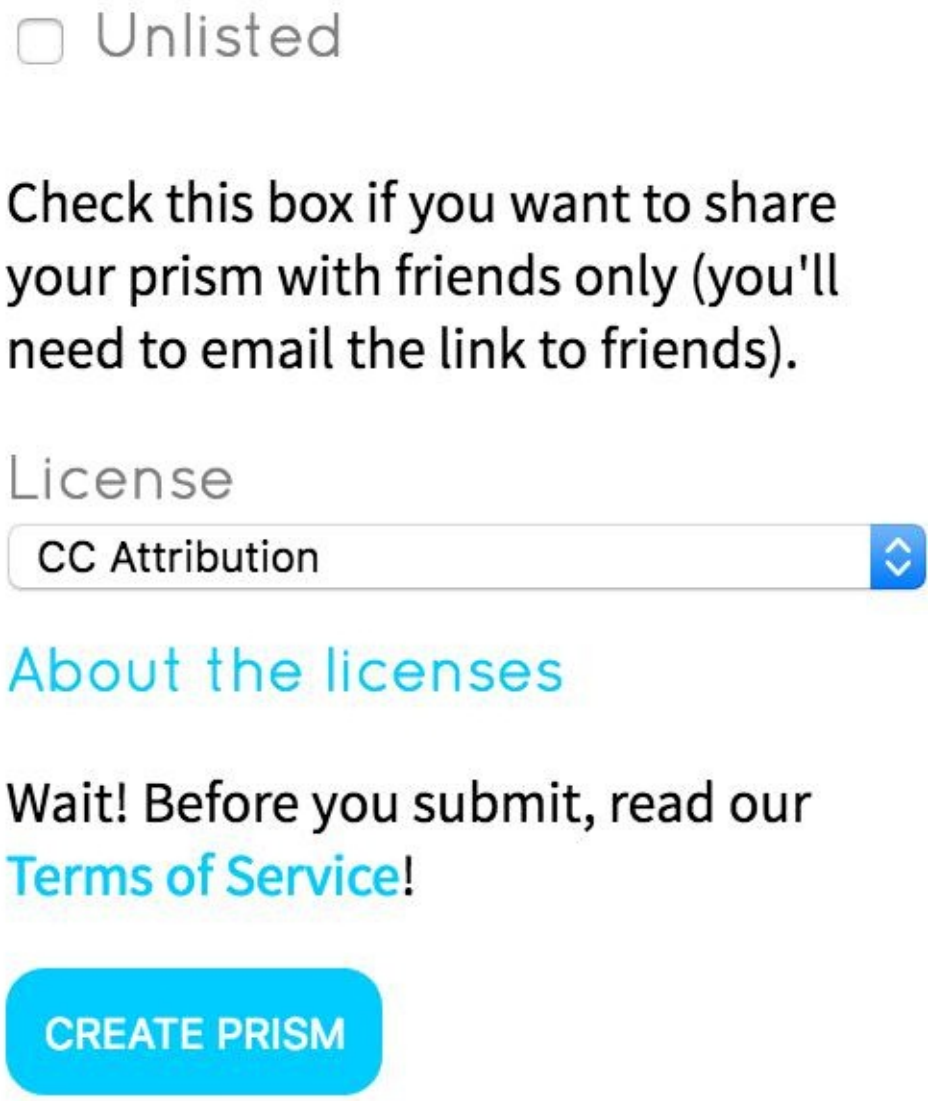

In this photo from the "Future Directions" page for Prism, you can see the prototype for another possible visualization that would shuffle through the various sets of highlights. Even without this animated interpretation, Prism allows you to get a sense of how a whole group interprets a text. The program collections you markings along with those of everyone who has ever read that text in Prism. We can begin to get some sense of trends in the ways that the group reads.

Prism was designed as a middle road between the two types of crowdsourcing projects that we discussed in the last section. By asking users to mark for a restricted number of categories, it can quantify those readings and visualize them in interesting ways. But it also asks for readers to actually read the text - interpreting a document along certain guidelines still asks readers to exercise the full range of their powers as thinking people. For this reason, the designers of Prism see it as **crowdsourcing interpretation**.

Prism offers a few options to facilitate group reading. Most importantly, it assumes very little about how its users will use the tool. Anyone can upload their own text as a Prism and, within certain guidelines, adapt the tool to their own purposes. When logged in, you can create a Prism by clicking the big create button to pull up the uploading interface:




You upload a text by pasting it into the window provided. Prism does not play well with super long texts, so you may have to play around in order to find a length that works for the tool as well as for you. The three facets on the right correspond to the three marking categories according to which you want users to highlight. The rest of these categories should be self-explanatory. Note, however, that you will only be able to give a short description to readers: your document and marking categories will largely have to stand on their own.

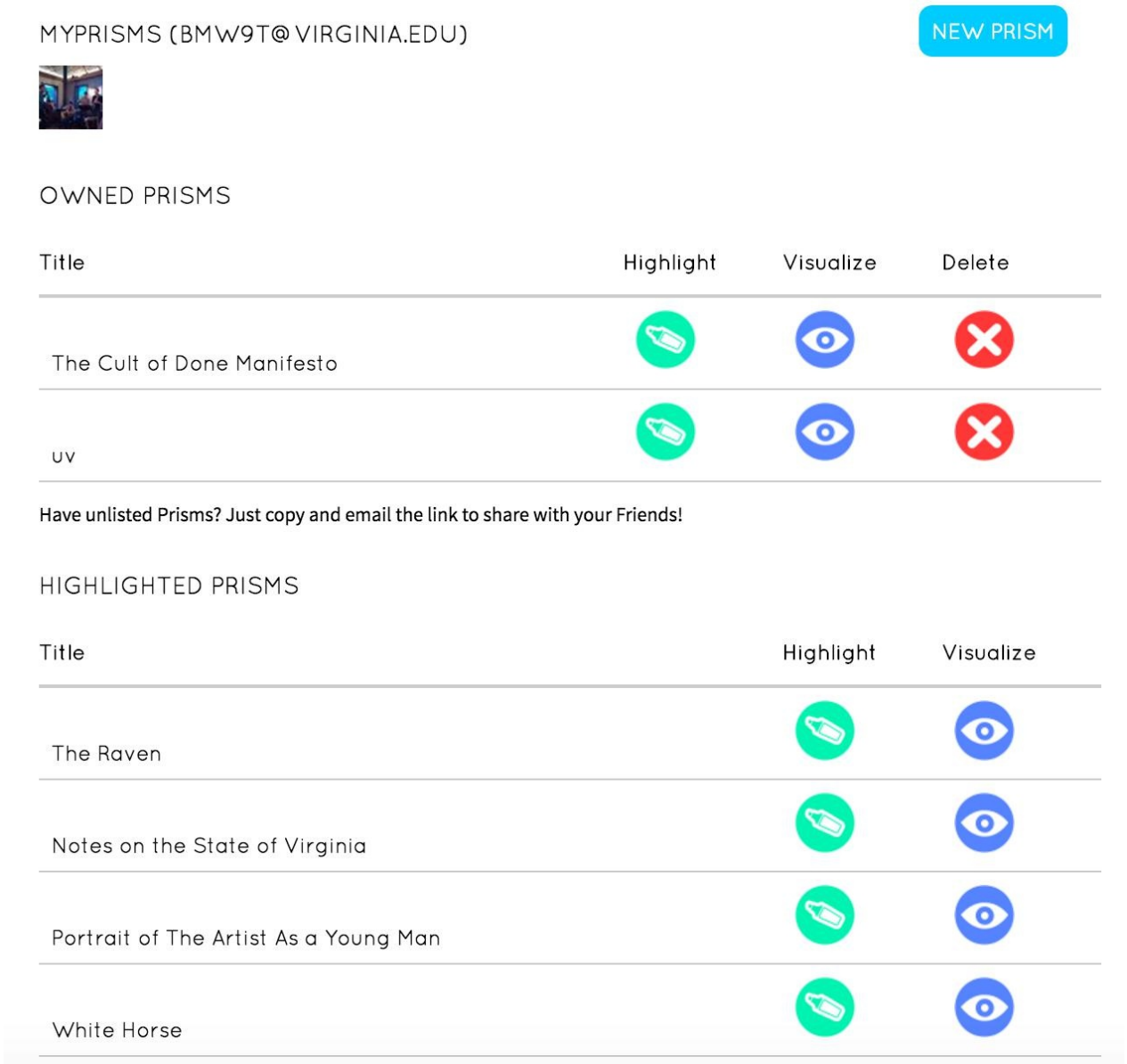



Below these main parameters for your text, you will be asked to make some other choices that may be less intuitive.

By default, Prism assumes that you want the text and all its markings to be made available to the public. Selecting **unlisted** will make your Prism private so that only to people to whom you send the URL can view it. Once you create the Prism, you will want to be extra certain that you copy that URL down somewhere so that you can send it out to your group.

Prism will also ask you what license you want to attribute to your materials. Many of the choices offered here are creative commons licenses, but you can also choose public domain or fair use depending on your needs. If you are unsure, you can always select no license, but it would be worth doing a little research about the materials you are uploading to find out their legal status.

Once you upload a text, the easiest way to find it is to go your personal page by clicking the "MYPRISMS" link from the top menu. In this profile page, you can easily access both the texts that you have uploaded as well as the ones that you have highlighted but that belong to others (quite handy if you lose the URL for an unlisted text).




With these tools, you can upload a range of texts for any kind of experiment. It is tempting to say that you are limited by your imagination, but you will run up against scenarios in which the parameters for the tool cause you headaches. That's OK! Take these opportunities to reflect:

- What will the tool not let you do?
- Can you imagine good reasons for these limitations?
- How would you design a different tool?

As you work through Prism, think critically about the concept of crowdsourced interpretation.

- Do you feel that this sort of work is fundamentally more empowering than the kind we saw with Typewright, Recaptcha, and Transcribe Bentham?
- Are there other, better ways of facilitating group collaboration, digital or otherwise?

# Exercises

# Exercises

Design a Prism game and answer the following questions. For your text, use an excerpt of at least 100 words from one of the following two writings by Jeremy Bentham on the panopticon:

- Letter VI
- Preface

1. What about this text are you interested in exploring? What is the interpretive intervention that you hope to make?
2. Who will you address it to? Why this particular group?
3. After designing your Prism get at least five people to read and mark the text for you.
4. What do you notice about the results? Anything interesting in the visualizations? The way people were interpreting the markings? The groups?
5. What limitations do you see with Prism? Can you imagine other ways of examining individual interpretations in the context of the group?

Implement your game and send us the link.

# Digital Archives

# Digital Archives

# Text Encoding Initiative

# Text Encoding Initiative

> Yet each man kills the thing he loves
>
> By each let this be heard.
>
> Some do it with a bitter look,
>
> Some with a flattering word.
>
> The coward does it with a kiss,
>
> The brave man with a sword!
>
> Oscar Wilde, *The Ballad of Reading Gaol*

This book studies texts and the things that computers can do with them. But, as you read along, you may notice that they cannot do all that much. Many of the methods that you will learn are simply sophisticated ways of counting words, whereas reading entails far more complicated processes of interpretation and analysis. When we read, we tend to skip to much more complicated understandings of a text:

- What does it mean?
- What elements of the text convey meaning? How do they do so?

Computers have a hard with abstract concepts like this. Computers tend to work in hierarchies and clear-cut structures, and, even then, they only know about those structures that someone has told them about. For example, if we were to say to a computer, "Hey! Find me the poem in this lesson!" It would have no idea what we were talking about: we have to find some way of telling the computer where it can find the poem. Right now it just thinks the text up there is no different from the other lines of text on this page. It's all just text.

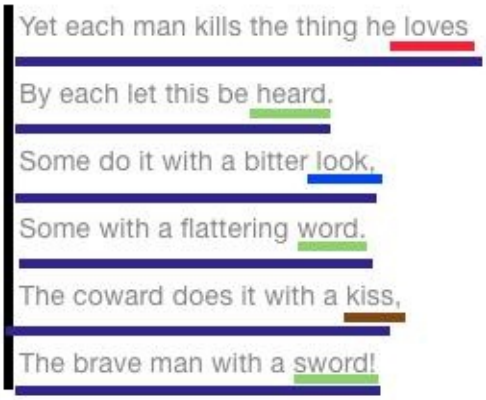
A computer program looking at the above passage from Oscar Wilde's *The Ballad of Reading Gaol* would, most likely, not even recognize those six lines as related in any way. Nor it would understand anything about how the internal components of those lines are connected to each other. There are a number of ways to represent such structural information, and we can get towards one that works for a computer by working through a system that you might use on your own when you read.

Think about all the annotations that you put on your own pages as you read them. If you are anything like us, your markings tend to be all over the place. But imagine if you were to systematically note certain structural features of the text. We can think of the Wilde passage, after all, as a series of nested concepts:

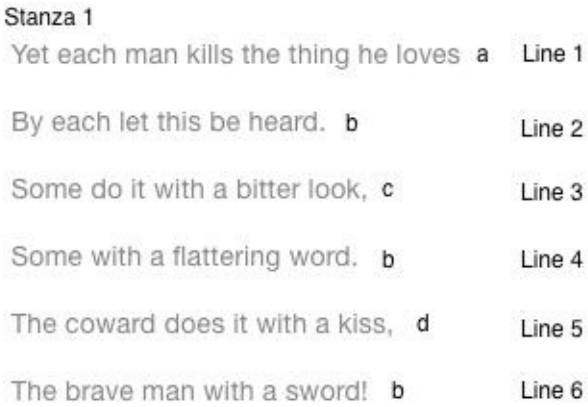
- There is a stanza.
- This stanza contains some lines.
- Each line has text.
- Some of that text contains a rhyme.

And we can represent it graphically, like so, where a black line denotes the bounds of the stanza, a

horizontal blue one represents the lines, and the rotating colors under the final words describe a rhyme scheme:




But you would probably need a moment to figure out what was going on if you came to this having not highlighted things yourself. We can do better. The following text annotations are a bit clearer and get closer to something we could understand without having any context. In fact, you may have worked with annotations like these before if you've taken a poetry class:




For a computer to understand this, we need an even more delineated way of describing the passage. We have to pay careful attention to **syntax**, the ways in which we mark particular things to provide information to the computer. Computers require very specific systematic guidelines to be able to process information, as you will learn in our chapter on "Cyborg Readers". For example, we would have to consistently use lower-case letters to represent rhyme schemes. And "line 1" to represent the first line of a poem instead of "line one." We need a clear and uniform way for describing the parts of the poem that never changes. Any variations from these rules would cause unwanted and unintended effects. Scholars have been working for years to develop such a system for describing texts in a way that can be processed by software. The **Text Encoding Initiative (TEI)**, the result of this work, is an attempt to make abstract humanities concepts legible to machines. If we apply TEI to the passage, it might begin to look something like this:

```
<lg>
  <l>Yet each man kills the thing he loves</l>
  <l>By each let this be heard.</l>
  <l>Some do it with a bitter look,</l>
  <l>Some with a flattering word.</l>
  <l>The coward does it with a kiss,</l>
  <l>The brave man with a sword!</l>
</lg>
```

Notice how our concepts like 'stanza' and 'line' have here translated into particular **tags**:

```
Our stanza becomes marked by <lg> (line grouping).
Our lines become marked with <l> (line).
Each set of tags has both an opening (<lg>) and a closing (</lg>) tag
Closing tags are almost identical to opening tags except for a forwar
Tags have intuitive relationships to the concepts that they represent
```

The opening and closing tags wrap around and determine the locations of particular structural elements for the computer: a line exists from here to there, a stanza exists from here to there, and so on. And take note of how certain tags can exist inside others. These framing elements help the computer understand the boundaries of the concepts we are describing, and they help to provide structure to the text. Think of TEI as a new layer that exists on top of the text. Words offer one layer of meaning, but we add new layers by marking the text up with these fixed annotations. This **markup** gives the computer (or future readers) more nuanced ways of understanding how the parts in a text relate to one another.

We can give further details to the poem. For example:

```
<lg type="poem">
  <lg type="stanza">
    <l>Yet each man kills the thing he loves</l>
    <l>By each let this be heard.</l>
    <l>Some do it with a bitter look,</l>
    <l>Some with a flattering word.</l>
    <l>The coward does it with a kiss,</l>
    <l>The brave man with a sword!</l>
  </lg>
  <lg type="stanza">
    <l> A short second stanza that we've made up.</l>
  </lg>
</lg>
```

Here we've added an extra stanza group as well as an outer tag to denote that this is, in fact, a poem. We also give **attributes** to certain tags to provide more information about them: type="stanza" tells the computer that the contents of this tag refer to a poem. Remember the nested hierarchy we talked about earlier? Notice how we represent it graphically by indentation. The outer poem element contains two stanzas, which contain some lines, and those have some text. You can run your eye down the text and see the structure. Some programming languages will actually error if you do not pay attention to such things. But, either way, it just helps us keep things clean and easy to read.

One last thing. Remember our rhyme scheme and line numbers? We can encode those too:

```
<lg type="stanza">
    <l n="1" rhyme="a">Yet each man kills the thing he <rhyme>loves</
    <l n="2" rhyme="b">By each let this be <rhyme>heard</rhyme>.</l>
    <l n="3" rhyme="c">Some do it with a bitter <rhyme>look</rhyme>,<
    <l n="4" rhyme="b">Some with a flattering <rhyme>word</rhyme>.</l
    <l n="5" rhyme="d">The coward does it with a <rhyme>kiss</rhyme>,
    <l n="6" rhyme="b">The brave man with a <rhyme>sword</rhyme>!</l>
  </lg>
```

We have added attributes to denote the line numbers for each line as well as the rhyme scheme, and then a new tag for each line denotes what word the rhyme is associated with. All of the tags used here

can be found in the tutorial on poetry on the TEI by Example page.

Once a text has been **encoded** in this way, it can be represented more easily in a digital form. This work may not necessarily actually make it *look* any different. But it does allow you to do new and exciting things to your work. TEI encoding can make it possible to provide nuanced digital editions of a text. We could actually say to a program, pull out all the rhyming words in a poem. Or make them appear differently on a webpage. Or change them all to "TEI is the best."

Let's look at an example of something that's got a lot of encoding already in it. Here is the TEI for this entry on a robbery case that mentions Jack the Ripper from the Old Bailey Online. You might not recognize a lot of the tags (there are *loads* of TEI tags), but the general arrangement of them should look familiar (full TEI here):

```
could see—the station is more than 50 yards from the public-house—I came back to the publichouse, put my head in and
saw him sitting there, and went for a policeman—I came back in four or five minutes; he was still sitting there—the
policeman called him out, and he was arrested—when he came out he did not say anything.</p>
<p>
<hi rend="smallCaps">
<persName id="t18881119-name-276" type="witnessName">
<interp inst="t18881119-name-276" type="gender" value="male"/>
<interp inst="t18881119-name-276" type="surname" value="CARTER"/>
<interp inst="t18881119-name-276" type="given" value="OLIVER"/>OLIVER CARTER</persName> </hi> (
<hi rend="italic">Policeman H</hi> 434). On 15th November, about a quarter to eleven, I was on duty in the
neighbourhood of Leman Street—I heard a noise, in consequence of which I went and saw the prosecutor, who made a
communication to me—I afterwards went with him to the Red Lion public-house—he went in first—I saw about 20 men in
there sitting round, the prisoner among them—the prosecutor said to me, "That is the man that robbed me," pointing to the
prisoner—I called the prisoner outside and told him what the prosecutor stated, and I said, "I shall take you into custody on
this charge"—he said, "I merely took him by the arm, and was going to take him to the station to give him into custody for
Jack the Ripper"—the prosecutor was quite sober.</p>
<p>
<hi rend="italic">Cross-examined.</hi> There were about 20 people in the public-house; no people outside—two men
followed to the station, no one else—there was no noise in the public-house—the prisoner did not say when I arrested him,
"It is a mistake."</p>
<p>
<hi rend="italic">The Prisoner's Statement before the Magistrate,</hi> "The man made a mistake altogether; I cannot say
nothing else."</p>
<p>
<hi rend="italic">Witnesses for the Defence.</hi> </p>
<p>
<hi rend="smallCaps">
<persName id="t18881119-name-277" type="witnessName">
<interp inst="t18881119-name-277" type="gender" value="female"/>
<interp inst="t18881119-name-277" type="surname" value="MURPHY"/>
<interp inst="t18881119-name-277" type="given" value="FLORANCE"/>FLORANCE MURPHY</persName> </hi>. I am a
stevedore of 162, Cable Street—on 15th November between 11 and 12, I was in the Golden Lion, when the prisoner and
```

Focus on what you do know: the tagging syntax should ring some bells. If you want to look up any of the tags, you can always check out the TEI guidelines. A lot of working with technology consists of not panicking when you see something unfamiliar and then looking up what you don't know. But we digress.

When you look at the public-facing version of the entry on the Old Bailey Online, almost all the tags disappear:

See original 

else but the four in the street—the prisoner struck me on my back with his hand, and with his leg too—it was dark, but I could see—the station is more than 50 yards from the public-house—I came back to the publichouse, put my head in and saw him sitting there, and went for a policeman—I came back in four or five minutes; he was still sitting there—the policeman called him out, and he was arrested—when he came out he did not say anything.

OLIVER CARTER (*Policeman H* 434). On 15th November, about a quarter to eleven, I was on duty in the neighbourhood of Leman Street—I heard a noise, in consequence of which I went and saw the prosecutor, who made a communication to me—I afterwards went with him to the Red Lion public-house—he went in first—I saw about 20 men in there sitting round, the prisoner among them—the prosecutor said to me, "That is the man that robbed me," pointing to the prisoner—I called the prisoner outside and told him what the prosecutor stated, and I said, "I shall take you into custody on this charge"—he said, "I merely took him by the arm, and was going to take him to the station to give him into custody for Jack the Ripper"—the prosecutor was quite sober.

*Cross-examined*. There were about 20 people in the public-house; no people outside—two men followed to the station, no one else—there was no noise in the public-house—the prisoner did not say when I arrested him, "It is a mistake."

*The Prisoner's Statement before the Magistrate,* "The man made a mistake altogether; I cannot say nothing else."

*Witnesses for the Defence*.

In order to make the text legible for readers, we very often hide most (or all) of the markup that is helping to present the document. This ensures that you can serve the needs of different audiences: some people might want to see the TEI for your text, but others might just want to be able to read it as normal. We have already talked a bit about the functions you can get from TEI, but, if they largely remain hidden, you might find yourself thinking that they might not be enough to warrant the amount of work that goes into putting together a TEI-encoded text.

Looking at the TEI tags for this document give you a sense of why you might encode a text even if not many people are going to look at the TEI. You'll see there are tags for the witnesses' names (both first and last) and their gender. Why might this be useful? Well, maybe the people who designed this site thought that users could conceivably want to search for the participants in a crime by name and by gender. That way you can ask all sorts of questions about what other crimes individuals witnessed, or whether men or women were more likely to be involved in or witness to particular types of crime.

Encoding is meant to convey abstract humanities concepts to the machine so that we can make better use of them in our digital work. Some of these concepts, like line breaks, might be pretty clear cut. Others might require a lot of interpretation. Even given the same set of tags and texts, two people could encode things differently. You can make a whole career off working with TEI, and we have just scratched the surface here. But encoding documents in TEI is an important step in preparing them for their lives as digital artifacts. Not all texts need to be encoded in TEI in order for them to be archived, but the vast majority of documents you will find in digital humanities archives have been encoded in this way. The process helps ensure that complex humanities data makes its way comfortably, ethically, and responsibly into the digital world.

## Further Resources

- Jacob Heil is great on the intellectual reasons for encoding in "Why We TEI."
- Ryan Cordell in "On Ignoring Encoding" gives a great defense of textual encoding as a fundamental part of digital humanities.

# NINES and Digital Archives

# NINES and Digital Archives

Our discussion of TEI has given you a sense of some of the work that can go into creating digital editions of texts. Indeed, encoding texts in this way is often the first step in a long process of preparing documents, historical or otherwise, for presentation and distribution on the web. In this section, we'll talk more about the stakes of putting digital collections like these online and help you understand some of the archives that are out there for you to use.

One good reason to put a collection online is be to increase access to the materials. After all, a manuscript kept in a museum requires that someone go to that location in order to read the document. An online version can reach a wider audience than a physical copy. However, putting materials on the internet raises a variety of legal and financial issues. After all, these digital resources require a great deal of time, funding, and energy to produce. Imagine you are the curator of an archive:

- Will you make your materials freely available to anyone with an internet connection?
- Will you require payment to see them?
- Why?

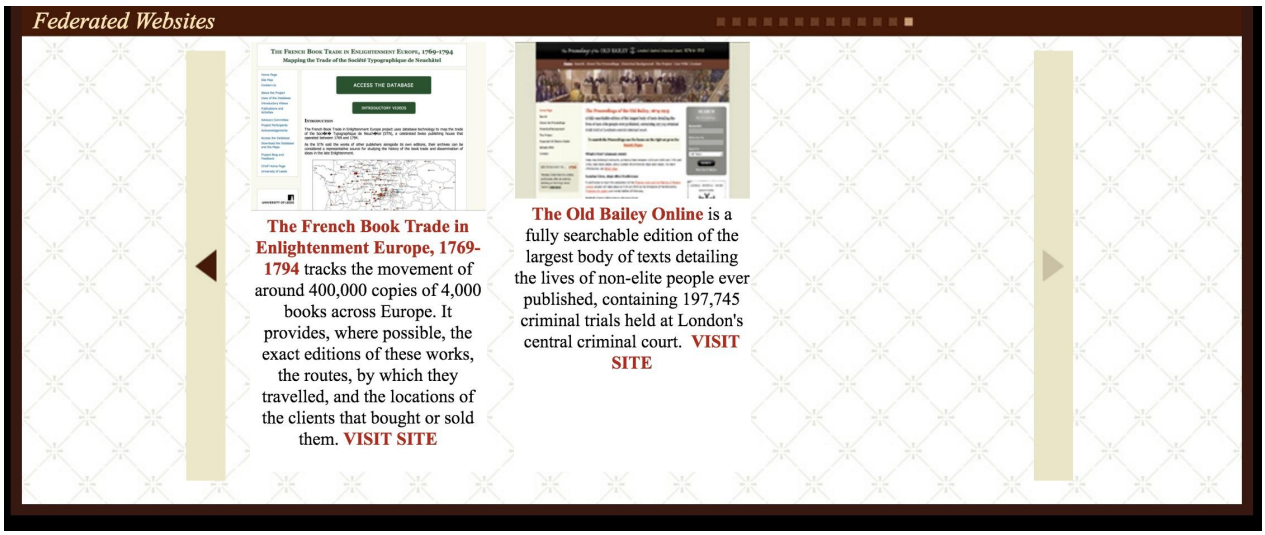
If you have ever tried to access a resource from an online newspaper only to be told that you need to subscribe to see their content, you have encountered such **paywalled** materials. Resources like these can be juxtaposed with **open access** materials. While there are different levels and variants, open access broadly means that the materials are available with little to no restrictions: you can read them without having to pay for them. For many, the choice is an ethical and a political one. But open access materials do raise serious financial questions:

- Keeping materials online requires sustained funding over time. How can open access work be maintained if they are presented for free?

Once materials are put online, it is possible to connect them to a wider, global network of similar digital materials. In the same way that a library gathers information about its materials to organize in a systematic way (more on **metadata** in our lesson on "Problems with Data"), scholars and archivists have to oversee this process for this networking to happen. For instance, technical standards shift (TEI tags can change over time), so archival materials require constant maintenance. If you have ever used a digital archive, you have benefited from a vast and often invisible amount of labor happening behind the scenes. The hidden work of gallery, library, archive, and museum (or **GLAM**) professionals ensures that our cultural heritage will remain accessible and sustainable for centuries to come.


The **Networked Infrastructure for Nineteenth-Century Electronic Scholarship (NINES)** is one such digital humanities organization that attempts to facilitate the archiving process by gathering archived materials pertaining to the nineteenth century. You might think of NINES as something like a one-stop shop for all your nineteenth-century archival needs. It gathers together peer-reviewed projects on literature and culture that different research teams around the globe put together; some focus on an individual author, others on a genre (periodicals or "penny dreadfuls") or a particular issue (disability or court cases). If you go to the site and scroll through "Federated Websites," you'll see the range of projects you can access from NINES, from one on the eighteenth-century book trade in France to another featuring the letters of Emily Dickinson. For the purposes of this class, you'll notice that some of the projects will be extremely useful to you, such as the Old Bailey Online, which contains trials records from London's central criminal court. Others, such as a project on the journals of Lewis and Clark, won't be relevant for this class, but might be for others you are taking.

You might also notice that NINES has a relatively expansive view of what the nineteenth century is, since this site includes projects that deal with the late eighteenth and early twentieth century. Historians often talk of the "long nineteenth century" as the period from the beginning of the French Revolution in 1789 to the outbreak of World War I in 1914. (In other words, historians of the nineteenth century like to claim big chunks of other people's time periods.)



Archives submit themselves for affiliation with NINES so that their materials can be searchable alongside other NINES sites, but first they must pass a rigorous process of **peer review** first. Academic journals rely on peer review to ensure that scholarship meets particular standards of rigor and relevance; it is a bit like quality control for scholarly writing. The peer review process typically involves submitting an article or book to a series of reviewers who write letters in support or rejection of the project and offer suggestions for improvement. The process is double-blind; the reviewers don't know who the authors are and vice versa. Should the piece pass, it moves onto publication and receives the explicit seal of approval from the publication.

| Resource | # of Objects |
|---|---|
| ▸ **Journals** | 698 |
| ▾ **Peer-Reviewed Projects** | 172 |
| British Women Romantic Poets | 3 |
| Carlyle Letters Online | 6 |
| Orlando: Women's Writing in the British Isles | 60 |
| ▸ **Romantic Circles** | 8 |
| ▸ **Rotunda Imprint, University of Virginia** | 3 |
| The Ambrose Bierce Project | 1 |
| The Correspondence of James McNeill Whistler | 6 |
| The Journals of the Lewis and Clark Expedition Online | 13 |
| The Poetess Archive | 2 |
| The Rossetti Archive | 9 |
| The Vault at Pfaff's | 43 |
| The Willa Cather Archive | 9 |
| The Yellow Nineties Online | 7 |
| William Morris Archive | 2 |
| ▸ **Other Digital Collections** | 66,069 |
| ▸ **NINES Exhibits** | 6 |

**Why go through peer review?**

Peer review sounds like a lot of work (and it is), but going through this process has benefits for you as an author. For one, it's a way to get suggestions for improvement. Scholars also see peer-reviewed projects as being more prestigious than non-peer reviewed works and, for the purposes of promotion, they "count" more than non-peer reviewed works. Peer review allows faculty members to assure their colleagues that their work is worthy of funding.

Digital projects, in particular, take an extraordinary amount of work and resources, so it makes sense that their contributors want credit for their work. But it can be difficult to evaluate something like an archive, since scholars are primarily trained to produce and evaluate secondary sources as opposed to primary-source repositories. Digital projects also require reviewers who understand not only the content but also the technical aspects of a project.

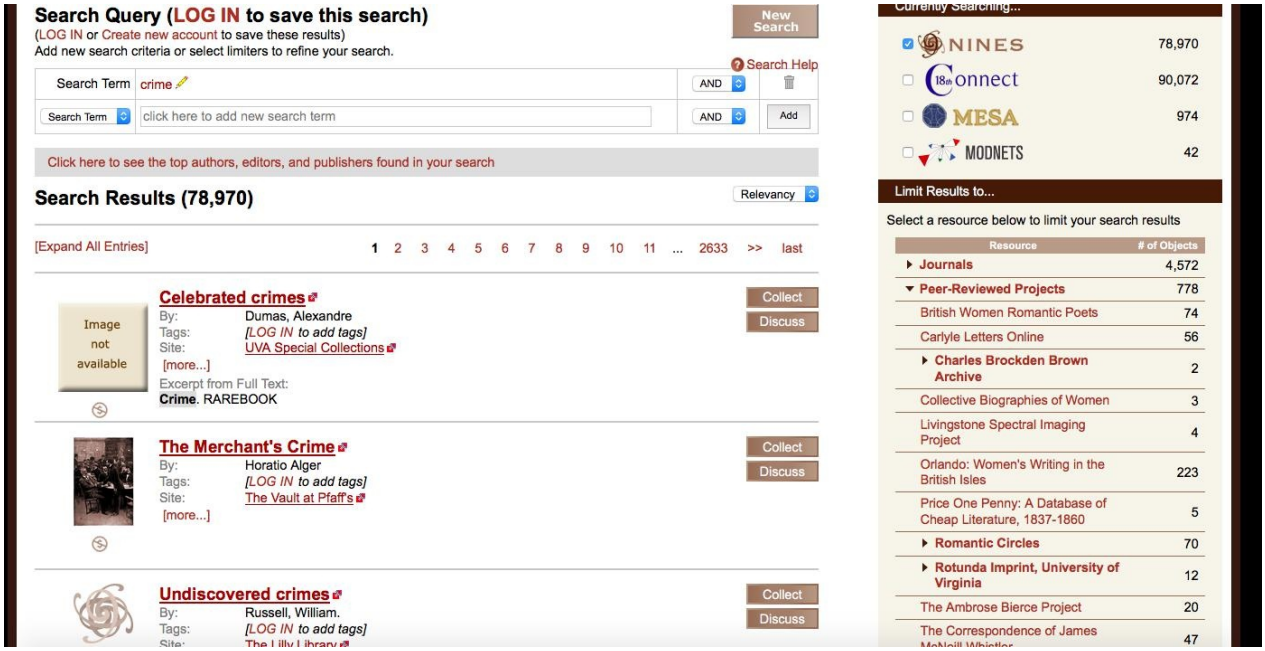
One of the early missions of NINES was to faciliate peer review for digital projects. By assembling digital humanities scholars that could evaluate digital archives and attest to their successes or flaws, project coordinators could better make their work available and understandable to colleagues who weren't working with digital material. So, say you worked on The Old Bailey Online and are up for a promotion at your institution; submitting this project to NINES for peer review is a way to make sure that your colleagues recognize the hard work you put into this project. Once reviewed, NINES makes the archival materials available for searching alongside other peer-reviewed projects. (You can see an example search of The Old Bailey Online here. Because the Old Bailey's archival materials are part of NINES, a search for 'old bailey' in NINES reveals objects not only in NINES, but also in a wide range of other archives.)

**What does peer review mean for you as a user of an archive?**

If you've made it this far in life, you've probably realized that you can't trust everything you find on the internet. In this case, knowing that something is peer reviewed allows you to put more trust in what you find on NINES than what you find elsewhere; you know that other scholars in the field have signed off on this material and think it is a worthy project.

**Why else should I use NINES?**

Beyond the fact that you can have a lot of confidence in the projects you find here, NINES is going to make it easier for you to find things. For one, you might not have known about all these different projects. NINES has also made sure that these projects "play nice" with each other (a.k.a. interoperability), which means you can find references to a particular topic or word across these projects with a simple search.




Doing a search for "crime" gets you all the references to this term in all of the different projects linked to NINES, saving you from having to search each individual archive.

One warning: only some of the results you get in the left pane are to material from the online projects affiliated with NINES. In other cases, NINES is searching library catalogs where the material is not available digitally. In this instance, if you wanted to read the first work, Alexandre Dumas's *Celebrated Crimes*, you would have to drive to Charlottesville and go to UVA's Special Collections Library.

- What archives do you use on a regular basis?
- What kinds of work do you imagine went into them?

# Exercises

# Exercises

- Imagine you are marking the following passage on prison life from The Dictionary of Victorian London in TEI. What elements would you tag or mark? (No need to actually look up the valid TEI codes for such things - you can just invent fake tags for what you would be interested in.)

---

Victorian London - Prisons - breaking windows to get into prison

WINDOW BREAKING

Sir, - Instances are now becoming more frequent of paupers preferring a prison to a workhouse, and resorting to the method of window breaking, as described in your police report of yesterday. Now, the law in its present state is merely an incentive to a repetition of the act; and, therefore, as it affords me no redress, I intend to take it into my own hands. I employ two porters on my premises, and have provided them with stout cudgels. If any pauper should deliberately break a large square of glass they will rush out, and thrash them most unmercifully. Where is the advantage in giving them into custody? By that means you confer a favour on the offender; and the very hour he is at liberty he will return and continue to repeat the offence until again incarcerated. It is no argument to tell us to use less expensive glass, as the pauper would soon find other means of accomplishing his object. What is required is this - and I ask the assistance of your all powerful pen in its favour - that a law should be passed condemning the perpetrator to a sound whipping and immediate discharge.

I am, Sir, your obedient servant, A CITY TRADESMAN.

*letter* in The Times, January 5, 1850

---

- If you could create an archive of some nineteenth-century materials, what would interest you and why?
- What legal or propietary issues would you have to sort out as part of the project?
- Who do you imagine would be interested in your archive?
- Would your site be open access or behind a paywall? Why?

# Data Cleaning

# Data Cleaning

# Problems with Data

# Problems with Data

So you have a text. You want to do something with it. It might be tempting to dive in and start using one of the tools in this book, but you should take a moment to examine the materials you are working with. Not all text is created equal, and your results can have real problems if you don't take care to examine the quality of the materials before you work with them.

The basic principle to remember is **garbage in, garbage out (or GIGO)**: you won't get good results unless you have good data to begin with.

## OCR

THESE EYES COULD READ THE INMOST THOUGHTS OF THE GUILTY!

Cool, canny, baffling, Sherlock Holmes was a figure to be reckoned with by the masters of lawlessness

# John Barrymore

*Idol of America and greatest actor of our time brings at last to motion pictures the most thrilling conception of all fiction*

*in*

# SHERLOCK HOLMES

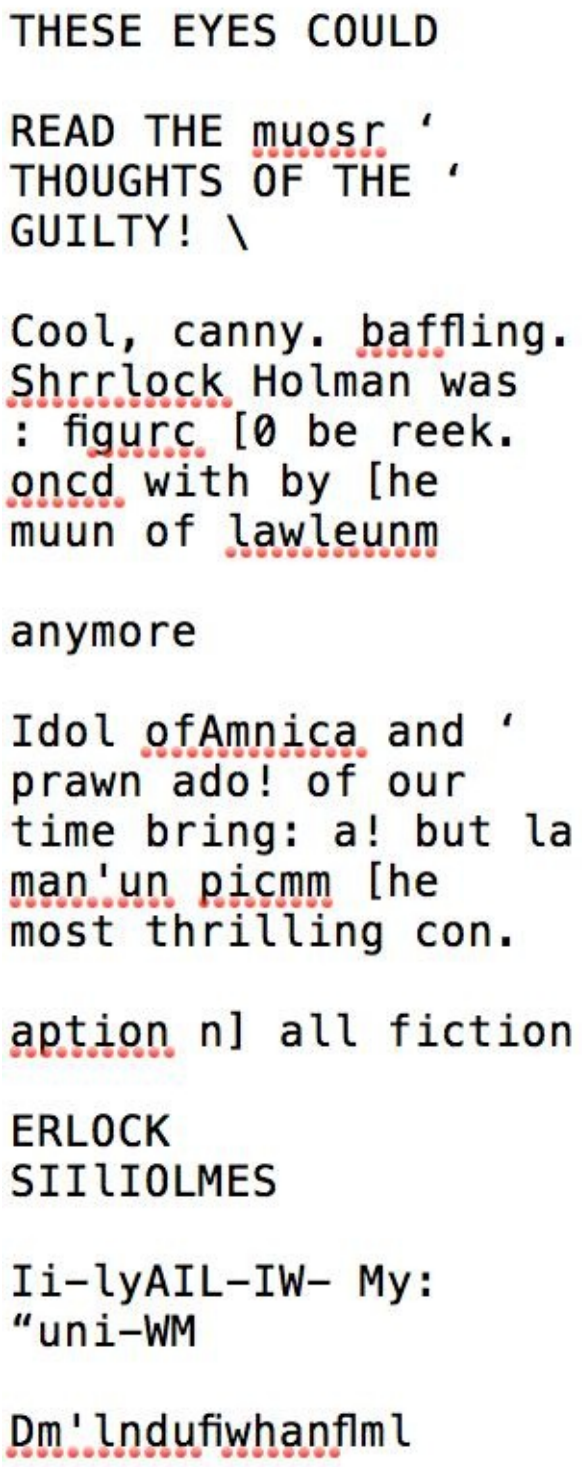
*Directed by* Albert Parker. *Adapted from* William Gillette's *stage play founded on Sir* Conan Doyle's *stories*

*Don't miss it when it comes!*

Take this image, drawn from a 1922 printing of *The Duluth Herald*, of a newspaper ad for the American film version of Sherlock Holmes.

By default, the computer has no idea that there is text inside of this image. For a computer, an image is just an image, and you can only do image-y things to it. The computer could rotate it, crop it, zoom in, or paint over parts of it, but your machine cannot read the text there - unless you tell it how to do so. In fact, the computer doesn't even really know that there *is* text there. As far as it's concerned, an abstract painting and an image like this contain the same amount of textual information. The computer requires a little extra help to pull out the text information from the image.

The process of using software to extract the text from an image of a text is called **optical character recognition** or OCR. We occasionally use OCR as a noun, as in "the OCR for that document is pretty poor" or as a verb, as in "we need to OCR this text before we can process it." There are many tools that can generate OCR for a text, and some of them are proprietary, meaning you have to pay for the abilty to use them. All of these tools are only so good at the process: what is easy for you requires a lot of computing power to carry out effectively.




Running this image through tesseract, a common free tool for OCR'ing text, we get the computer's best garbled attempt at translating image into text (at right).

The material here is still recognizable as being part of the same text, though there are obvious problems with the reproduction. At first blush, you might think, "This should be easy! I learned to read a long time ago. I can even read things written in cursive! Why does the computer have such a hard time with this?" This is one of those instances where what is no trouble for you is much harder for a computer. Humans are great at pattern recognition, which is essentially what OCR is. Computers, not so much.

OCR'ing text is actually a pretty complicated problem for computers. WhatFontis.com lists over

342,000 fonts, and this count only appears to include Western fonts. A single word will look slightly different in each font and at each size. And that doesn't even begin to account for hand-written text or text that has been partially damaged: even a slight imperfection in a letter can complicate the scanning process. The process is complicated and takes a lot of work: even the most expensive OCR software is prone to errors. If you see clean text transcriptions of an image online, odds are high that a human cleaned up the OCR to make it readable.

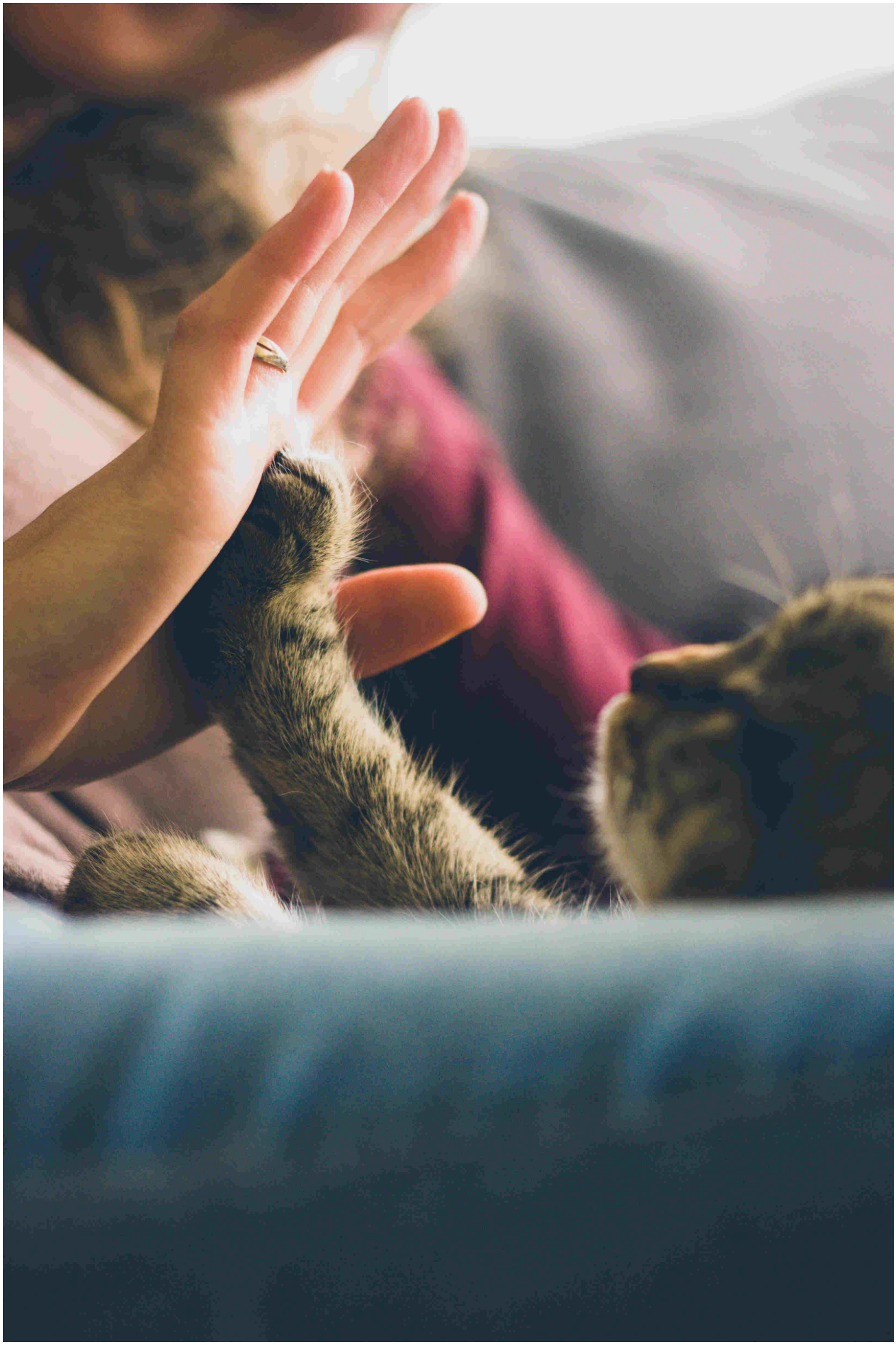
## Data Cleaning

Let me say it again, computers cannot infer. Imagine this scenario:

We're going to count to ten!

1,2,3,4,5,6,7,8,10

You probably meant to have a 9 in there, and a human reading it would most likely know that there was a mistake. But the computer will have no idea that you accidentally left out a number. You would have to specifically tell it to account for such errors. This simple fact about computational logic becomes a big problem in the humanities, because humanities data is *messy*. To see what we mean, go check out the Wikipedia section on Sir Arthur Conan Doyle's name. We will wait. Here is a picture of a cat in the meantime. Imagine it's a cat high fiving you when you clean up some data.



Did you read it? Promise?

Doyle has a complicated naming history, to say the least. Now imagine you are putting together a database of authors. You get to Doyle. How will you save his name? We can think of a number of possibilities:

```
Doyle Arthur Doyle
A.C. Doyle
Doyle, A.C.
Doyle, Sir Arthur
Doyle, Sir Arthur Conan
Sir Arthur Conan Doyle
```

You can probably imagine others. All of these are technically correct, and they might serve your purposes just fine. But you need to be consistent. Remember how computers cannot infer anything? Imagine this as part of your database of authors:

```
Author Name
---
Austen, Jane
James Joyce
Arthur Conan Doyle
```

You are working with a number of formats:

```
Author Name
---
Austen, Jane: last_name, first_name
Arthur Doyle: first_name last_name
```

A computer program would need a way to understand what you are giving it, something like:

1. Look at this 'Author Name' database.
2. Each Author has a line of its own.
3. Get the Author's name.

This data would cause all sorts of problems with the third step. To begin, how does the computer get the names? There are two options here:

- Look at the line for a comma. Before the comma, you will find the last name. After it, you will find the first name.
- Look at the line for a space. Before the space, you will find the first name. After it, you will find the last name.

The former is the more common way of representing data like this. Using commas to denote the different pieces of data is so popular that the format has its own name: **comma seperated value** or **csv**. It has an advantage over the second format that breaks apart data based on spaces:

```
Author Names
---
Austen Jane
Arthur Doyle
Arthur Conan Doyle
```

If we used spaces to denote breaks between first name and last name, Arthur Conan Doyle would

cause our program to error. It would likely interpret 'Arthur' as the first name and 'Conan' as the last name. 'Doyle' would be an unkown. Reformatting this as a csv allows us to handle Conan Doyle's full name:

```
Author Names
---
Austen, Jane
Arthur, Doyle
Arthur Conan, Doyle
```

The next problem should be obvious: Jane Austen is in a last_name, first_name format, while the others are in the reverse. So our final version of this dataset would look like this:

```
Author Names
---
Austen, Jane
Doyle, Arthur
Doyle, Arthur Conan
```

We might go further to associate Arthur Doyle and Arthur Conan Doyle as being representations of the same person, a process known as **authority control**. A common way of referring to data that contains inconsistencies and/or errors is as **dirty data**. To keep the metaphor, then, the process of revising data to remove such problems and prepare it for use is called **data cleaning**.

# Metadata

If you have ever searched for a book using a library search interface, you have interacted with metadata categories. **Metadata**, in its most basic sense, is data about data. A text, after all, is more than just the words on the page. We have a whole range of other information that we use to describe the document. The author, its date of publication, its publisher, its copyright status, etc.: we might care deeply about these pieces of information, and we might want you to use them for particular analyses. These categories allow us to do things like search for books with particular titles from particular time periods. In our previous example, we were actually working with metadata without realizing.

```
Author Names
---
last_name, first_name
Austen, Jane
Doyle, Arthur
Doyle, Arthur Conan
```

We have two metadata categories here: last_name, and first_name. Each are separated by a comma. We might even think of author_name as being its own metadata category for someone else's list of books! Databases are really these sorts of things at their heart: data and metadata, organized in systematic ways to make them easily usable.

Imagine you have started to put together your own table of author names and you notice that your neighbor is putting together one of her own. You want to be able to compare notes and, even more, you want to combine lists. It should be obvious that you will have real problems if you organize things as "first_name last_name" and she organizes things as "last_name, first_name". You would need to do a lot of extra work to merge your two lists. It would have been easier if you were working with an accepted standard for how author names should be listed.

Such metadata standards exist, and a lot of work goes into maintaining them (check out Dublin Core if you are interested in learning more). These standards ensure that anyone producing a new dataset creates work that could easily translate and communicate with other systems. They ensure that your local library's data could eventually be drawn into the Digial Public Library of America and made available on a large scale. The process might seem easy with this basic author name example, but imagine trying to coordinate such metadata standards for all people working on all types of cultural objects, all over the world. The work never ends.

You can fall down a deep pit looking at all the different metadata standards and their uses. For now, we just want you to be familiar with the concepts.

## Further Resources

- Chris Woolford has a more detailed explanation of how OCR works at explainthatstuff.com.

# Zotero

# Zotero

You might be thinking to yourself, "I can't believe you made me read an entire section on data cleaning." Sorry you feel that way. Here is a dog in a blanket:

Let's just pretend you said something on the order of, "metadata is all well and good, but how am I going to use this in my everyday life beyond library searches?"

We're glad you asked! In fact, metadata is something you deal with all the time. It is all around us, structuring our lives. If you go to a store and go to a particular section to grab cereal, metadata about the foods on that aisle has helped organize them and guide you to them. If you look through your phone to see that you have a missed call, you'll be presented with information about the caller, the call duration, the number, etc. All metadata. You get the idea.

For a more concrete example relevant to this book, imagine that you were asked by your instructor to write a paper on the Sherlock Holmes story "A Scandal in Bohemia." You will probably need to include a bibliography. Something like this:

Doyle, Arthur Conan. “A Scandal in Bohemia.” Sherlock Holmes: Selected Stories. Ed. Barry McCrea. Oxford: Oxford University Press, 2014. Print. Oxford World’s Classics.

Hopefully you see where we're heading: that citation is composed of nothing but metadata:

```
Book Section
Author: Doyle, Arthur Conan
Title: A Scandal in Bohemia
Book: Sherlock Holmes: Selected Stories
Editor: Barry McCrea
Location: Oxford
Publisher: Oxford University Press
Date: 2014
Format: Print
Series: Oxford World’s Classics.
```

The citation is just metadata, organized according to a pre-arranged format: the MLA 7th Edition, in

this case. Anytime you create a citation, you are just organizing the metadata for an object in an agreed-upon format. But the process is very tedious, as we're sure you've noticed over the years. Fortunately, if you've been following along, you know that computers are exceptionally good at executing tedious tasks just like these.

We know the formula for an MLA citation looks like this:

Author. Title. Book. Editor. Location: Publisher, Data. Format. Series.

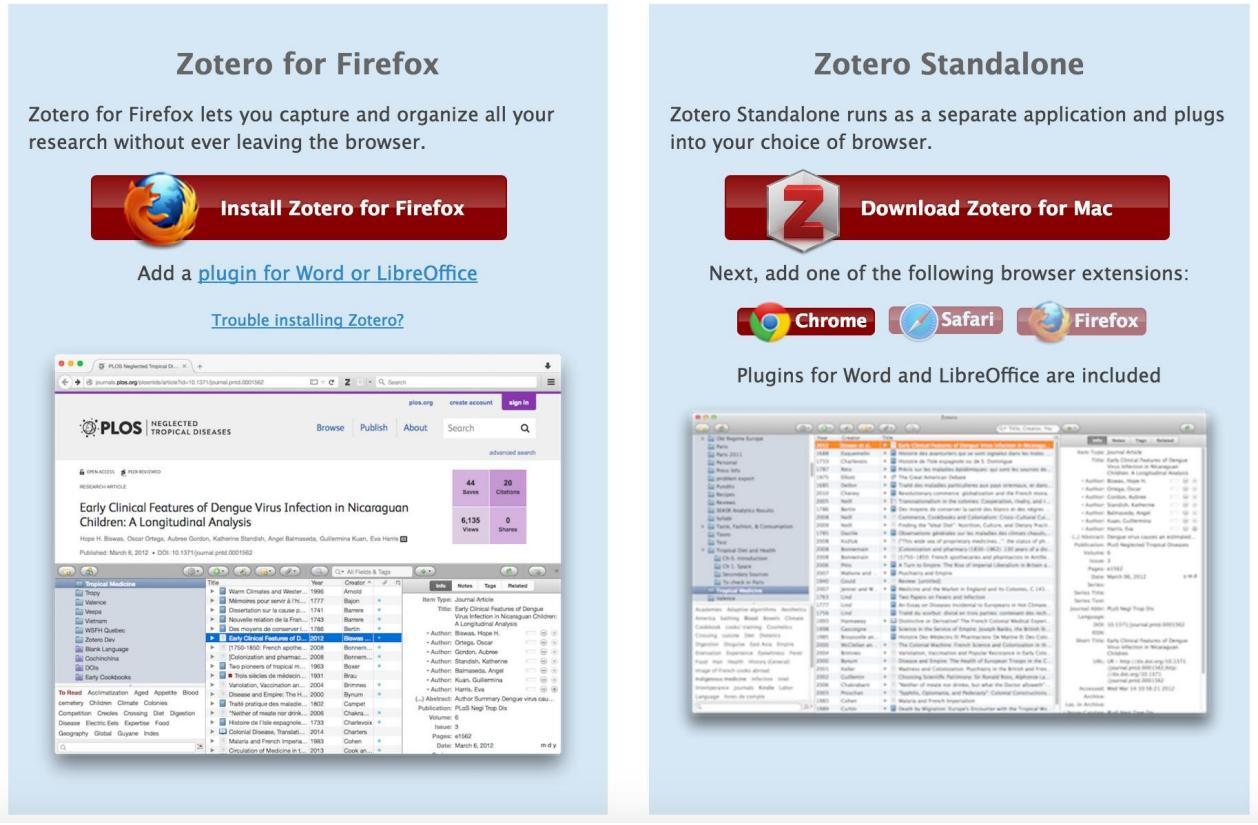
The formula will vary slightly depending on the type of item you are citing, but, if we have a full metadata listing for each text, we could imagine a tool that would fill in the metadata and format it to our recipe. We could imagine software that would spit out a bibliography for us.

If you have ever used something like easybib.com to produce a bibliography, you have done exactly the process that we are describing. You fill in the metadata you want, and easybib spits out a citation.

You might have felt like you were cheating when using a tool like this as a shortcut for your citations, but we're here to let you in on a secret. Your instructors rely on similar tools. If you think putting together a bibliography for your 5 page essay is irritating, imagine doing one for a 300 page book manuscript. And at the end of the process, your publisher decides that you have to use Chicago style (footnotes) rather than MLA (in-text citations). Do you want to convert them all by hand? Neither do we.

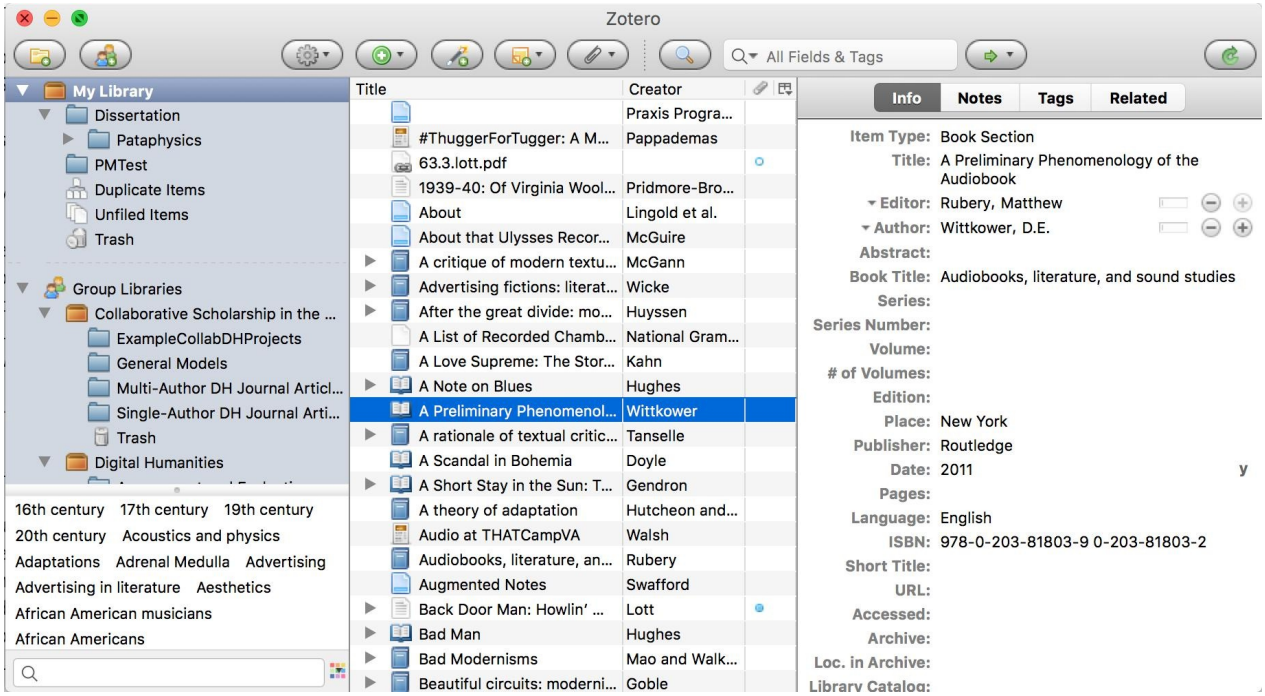
To deal with citation situations like this, the academic community has developed its own tool: Zotero. Zotero can do everything we've described so far in a snap, but it can also do so much more. Spending a few minutes learning it will save you dozens if not hundreds of hours in your writing life. Seriously.

First visit the Zotero download page. You can run Zotero out of Firefox, which allows you to capture and manage your citations without ever leaving the browser. But we like to separate the process of collecting citations from managing and using them. Download "Zotero Standalone" from the right window pane:



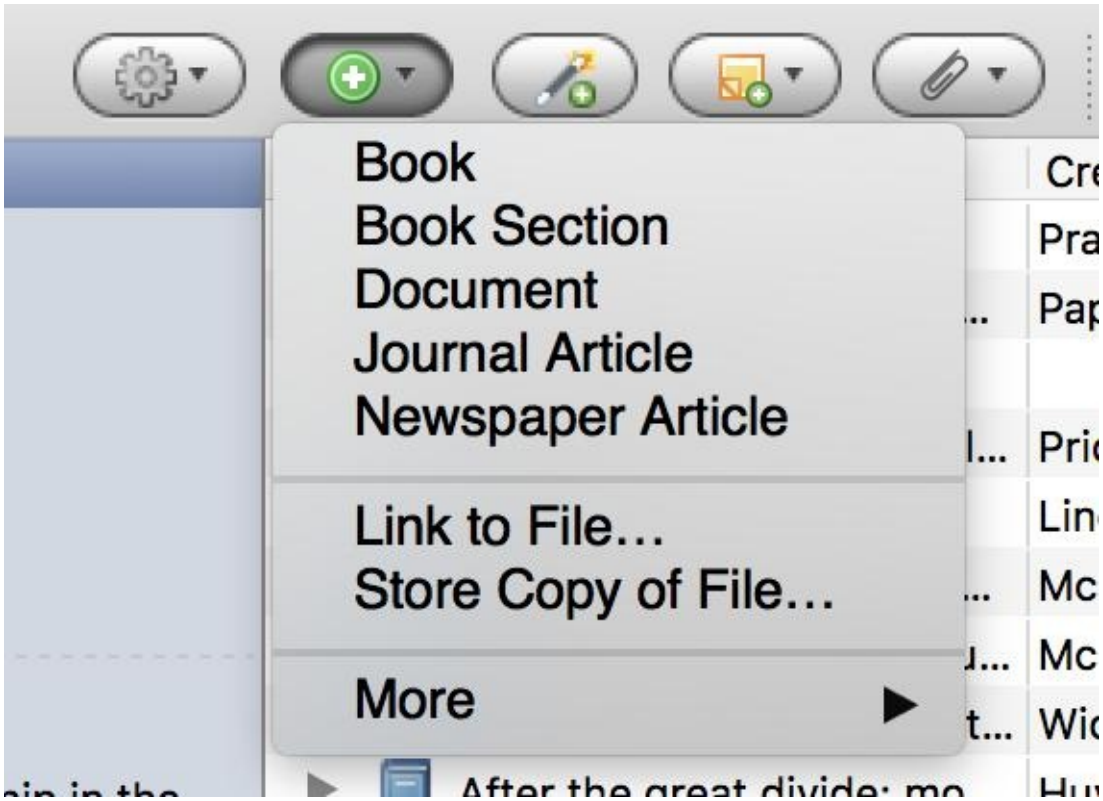

This will download an application to your desktop that, if you're like us, you'll want to put in a place where you'll have quick and easy access to it. While you're at it, you will need to add at least one of the browser extensions by clicking on the button in the same pane from the downloads page. We recommend you just add the extension to every browser that you have on your computer: these little downloads are what will allow you to pull citation information down off a webpage.

Once you download all those, open up Zotero Standalone. It should look something like this:



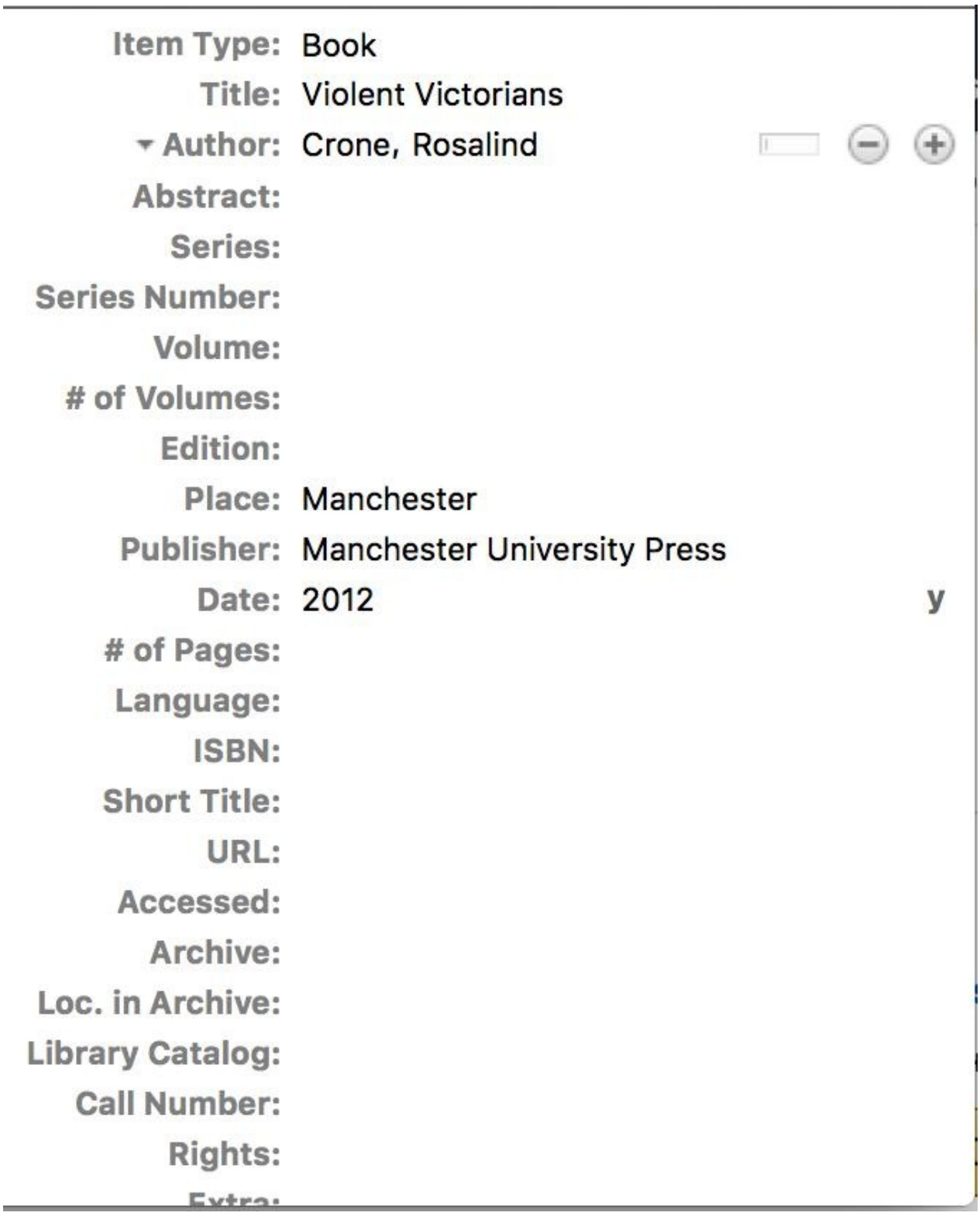

Your Zotero installation and library will look different from Brandon's, because his is full of materials related to things he has written. It will probably look pretty empty, since you haven't filled it with any

items just yet. Let's show Brandon what's what by grabbing some metadata to store it in Zotero. We will do this three different ways.

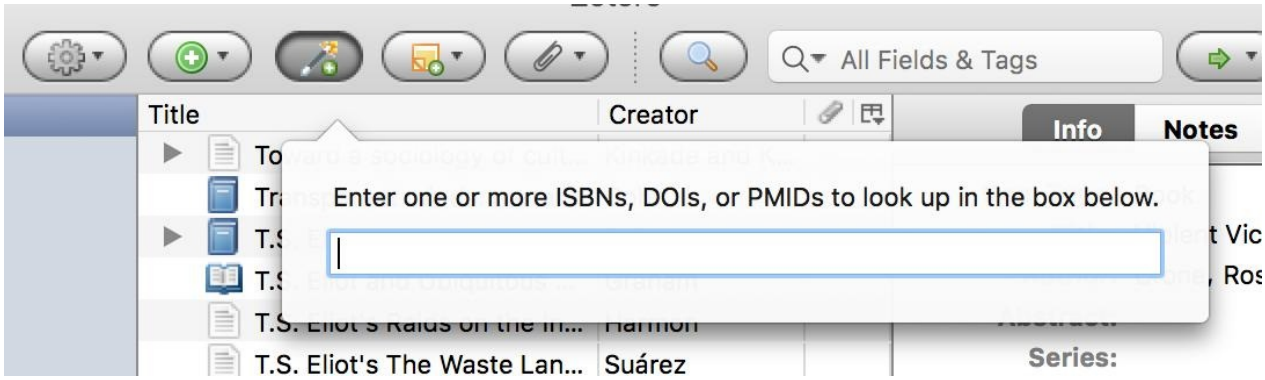



First, let's enter information manually. Let's pull out our copy of Rosalind Crone's *Violent Victorians* (you can find the relevant metadata on Amazon if you don't have your own copy. By clicking on the plus sign at the top, you can select the type of object you are adding to your collection. This is a book, so let's select that.

Doing so will shift your Zotero pane so that you can enter the metadata for Crone's book on the right. Go ahead and do that. Notice how you can have some categories be empty - Zotero is not picky. It has all those extra sections so that it can fit a variety of different use cases, but not every object will use every metadata category. When you're done, you should have something like this.




Zotero now knows about our citation, and we could use it for any number of things. But before we move on, let's cover two other ways to add citation information. Every book is given an an identifying number, an **International Standard Book Number (ISBN)**, and we can grab metadata using this thumbprint. This number is unique to every book. Zotero can map these numbers to their associated metadata. Clicking on the magic wand at the top of the Zotero Standalone pane will give you a place to enter an ISBN:



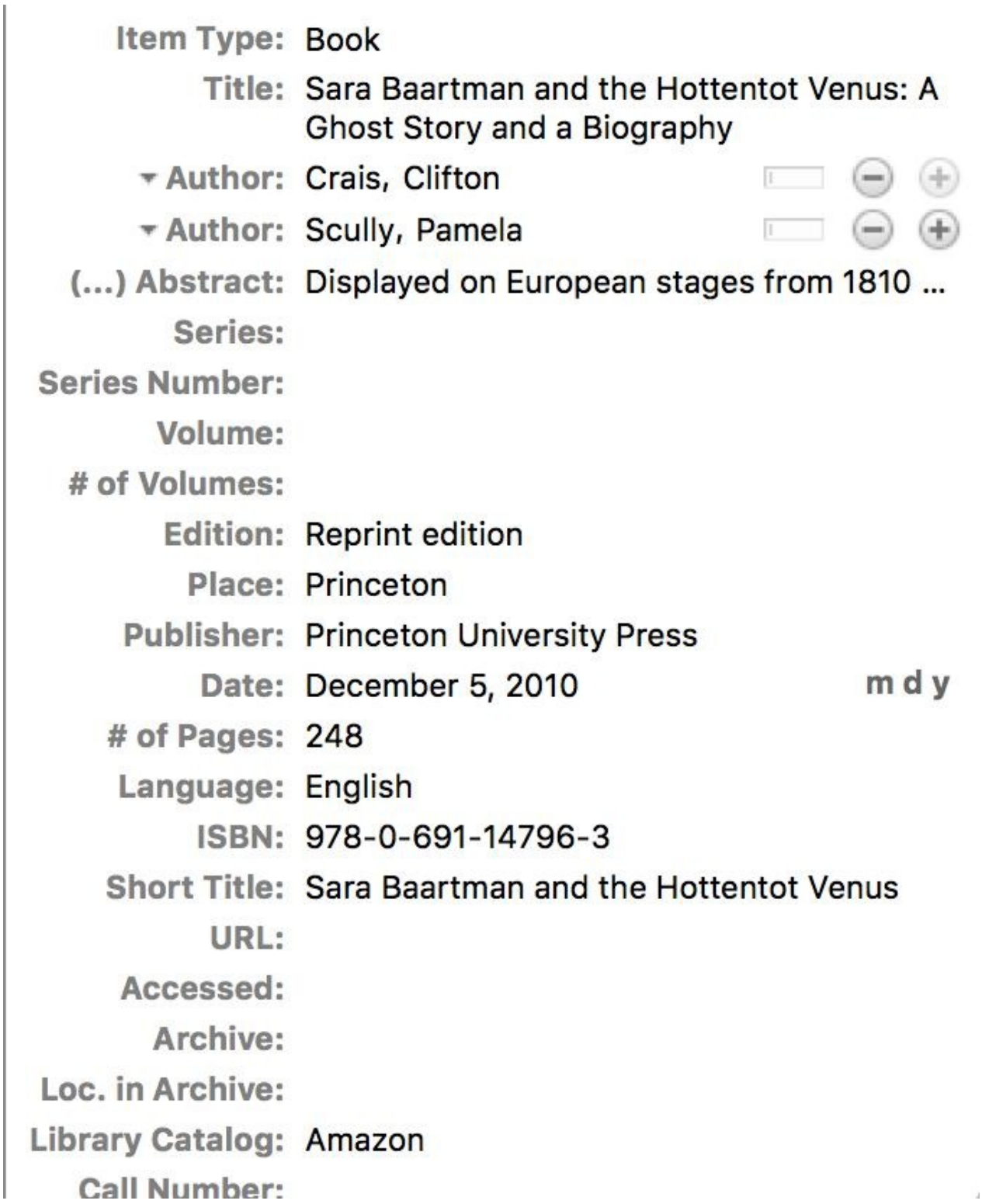

Try entering this one: 0520221680. Zotero should automatically go out and grab the metadata for its associated book: *Spectacular Realities* by Vanessa Schwartz. The result might not look perfect, and you might have to tinker with it to make it look like what you want. But you should wind up with another decent looking set of metadata for a second source:

| | |
|---|---|
| Item Type: | Book |
| Title: | Spectacular realities: early mass culture in fin-de-siècle Paris |
| Author: | Schwartz, Vanessa R. |
| Abstract: | |
| Series: | |
| Series Number: | |
| Volume: | |
| # of Volumes: | |
| Edition: | 1. paperback print |
| Place: | Berkeley |
| Publisher: | Univ. of California Press |
| Date: | 1999 |
| # of Pages: | 230 |
| Language: | eng |
| ISBN: | 978-0-520-22168-0 |
| Short Title: | Spectacular realities |
| URL: | |
| Accessed: | |
| Archive: | |
| Loc. in Archive: | |
| Library Catalog: | Gemeinsamer Bibliotheksverbund ISBN |
| Call Number: | |
| Rights: | |

Magic! But wait - there's more. Visit the Amazon webpage for *Sara Baartman and the Hottentot Venus: A Ghost Story and a Biography*. If you pay careful attention to your toolbars at the top of the webpage, you may have noticed a new one for Zotero (fourth from the left in this image from Brandon's computer).


By default, Zotero just assumes you are trying to grab the webpage itself. When you visit a page with

a citation you can download, however, the Zotero icon will change accordingly as it recognizes the metadata embedded in the page. In the image above, the Zotero icon appears as a book because it knows that it is looking at a book's metadata. Zotero will suck down the metadata on the page and store them in your Standalone App so that you can use them later. Magic!



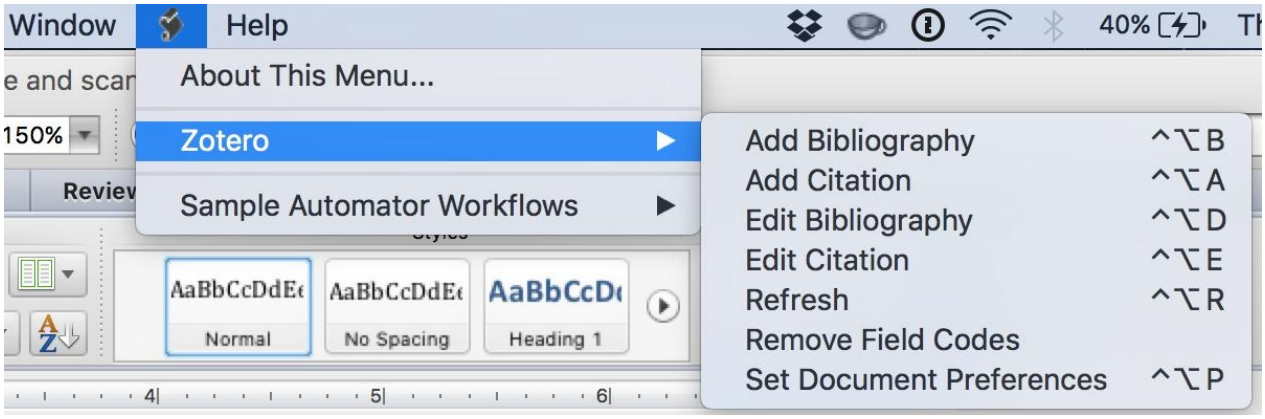

Well, not quite. We won't get lost in the weeds of explaining the technical details of how this works, as that is a subject for a different class. For now, just know that Zotero is interacting with hidden information on the webpage that provides information to programs like it. The average user never knows that these webpages contain information like this, but Zotero can leverage that invisible data into powerful content to make your life easier.

That being said, you need to remember the principle of garbage in, garbage out. In some cases, the webpage you are grabbing the citation from might not have complete (or entirely accurate) metadata. In that case, you will need to edit the citation in your Zotero library by clicking in the relevant fields, adding what's missing, and/or correcting any mistakes. Alternately, the format that libraries use to store metadata might be different from typical citation formats. Go back up to the record for *Spectacular Realities* and look at the title. You'll notice that the capitalization is different from what you might expect: the first word is capitalized, as are any proper nouns ("Paris," in this case). But if you were using this in a bibliography or citation (which we'll show you in a second), you'd want to correct the capitalization. You would also need to change "Univ. of California Press" to "University of California Press." This is all to say that although Zotero makes things much, much *easier*, it doesn't necessarily make them *automatic*.

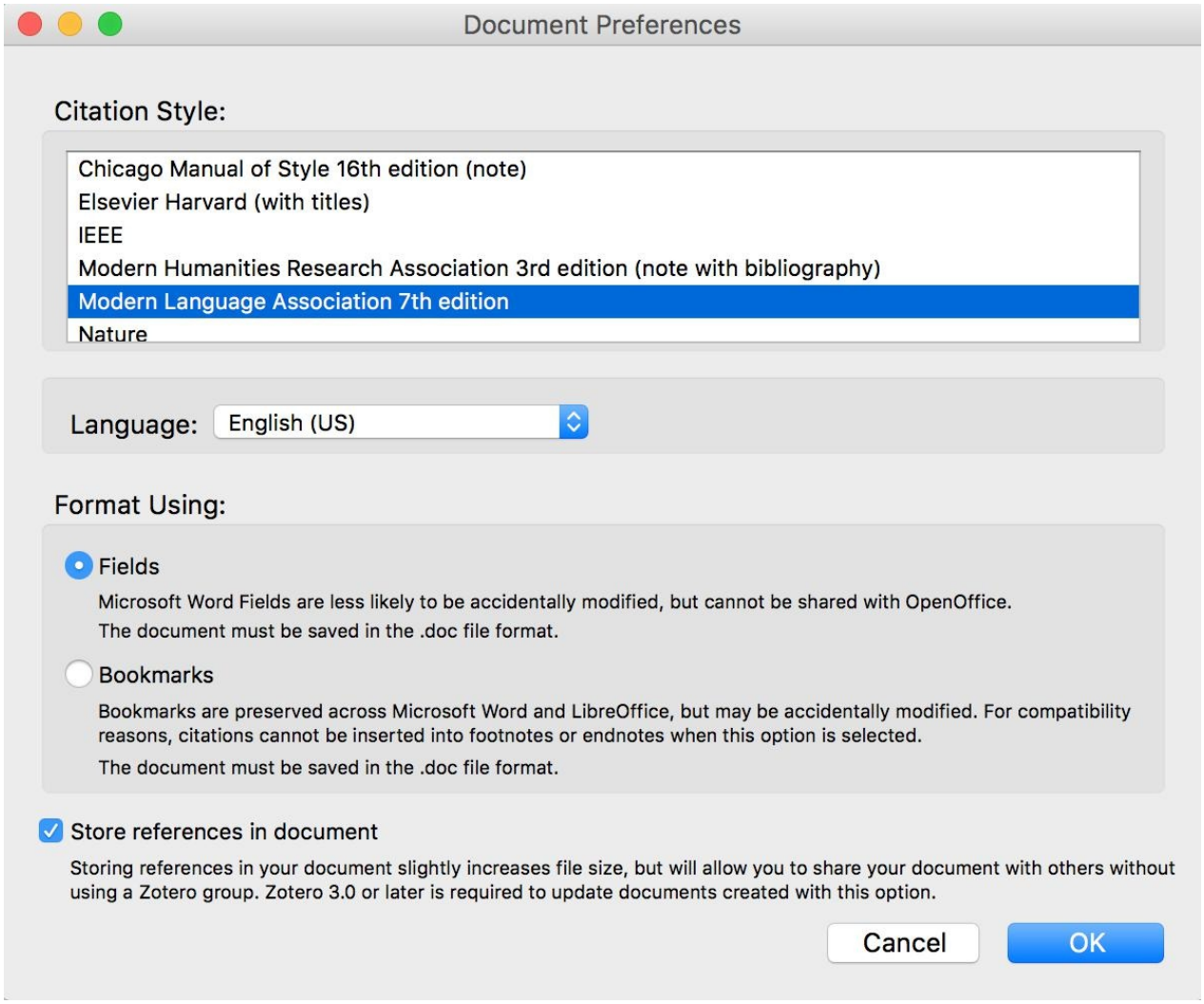
## Zotero for Citation Management in Microsoft Word

Now that we have our metadata, the fun begins. If you use Microsoft Word, strap in and buckle up. Go to the Zotero menu and select Preferences. From the 'Cite' menu, Install the Microsoft Word Add-in. Doing so will add a special 'Zotero' menu to every Microsoft Word document that you open. Now let's open a new Microsoft Word document and check out the new menu.

[Note: The following screenshots and narrative were written for Microsoft Word 2011, and later versions will vary slightly. Microsoft Word 2016, for example, changes the Zotero integration. In this later version you will be looking for a menu called "add-in's," and the keyboard shortcuts below will not work by default. The Zotero forums provide a tutorial for adding keyboard shortcuts manually. There can also find a number of screencasts to walk you through integrating with other Word versions.]

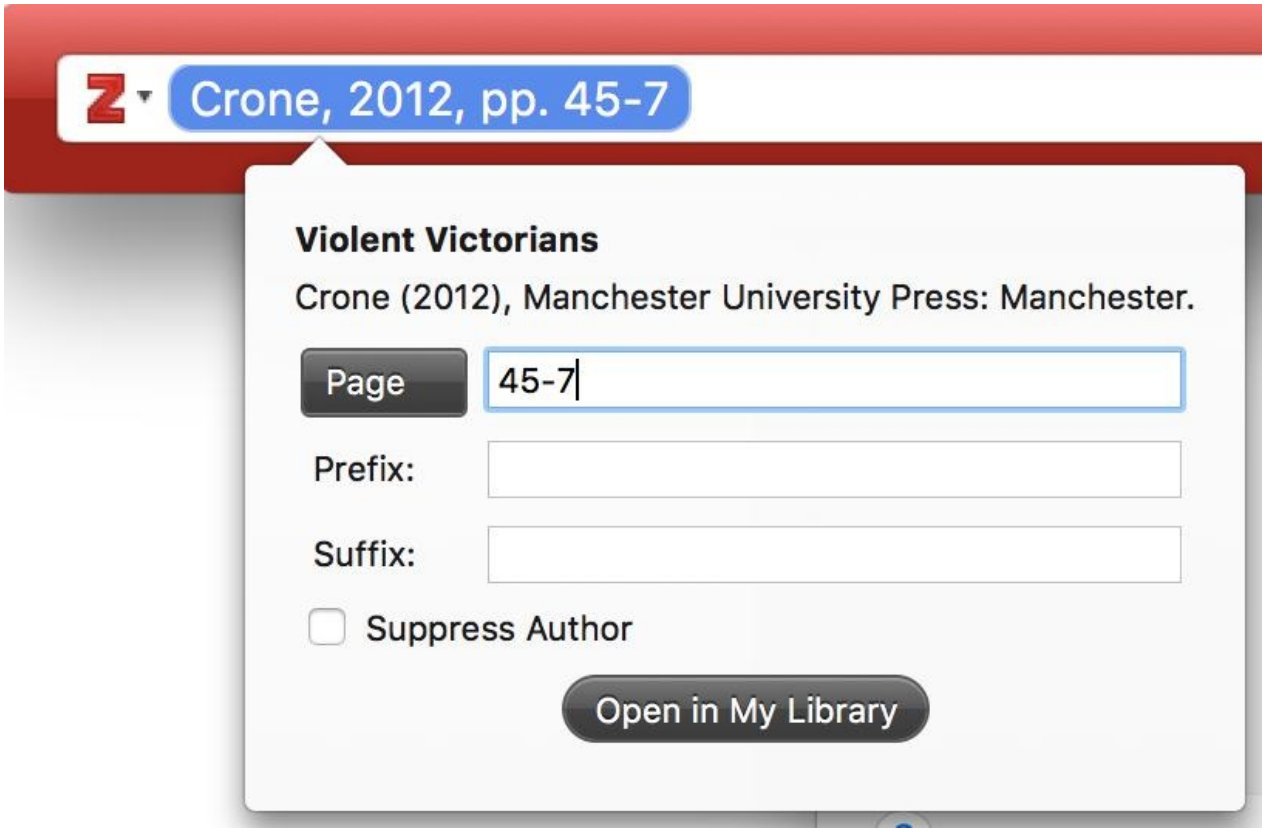



There are lot of options here, but the most important for right now are 'Add Bibliography' and 'Add Citation'. First, click 'Add Citation.' You will need to select a citation style since there is none associated with this document yet. Let's suggest MLA 7th Edition because Brandon is a literary studies person. It will make him happy.




Next pops up an input field that asks you to give some information so that it can locate a citation for you. Typing in 'Crone' will allow Zotero to recognize the author for the book we inputted and bring up the metadata we want. Click it to accept.


The input field now shows how the citation will show up. In most cases, however, we want to customize it. To do so, click on the citation to bring up some more options. Here you can add page numbers or, importantly suppress the author name depending on whether you only need the page numbers themselves. Let's give this entry the page numbers 45-7. Hit return to accept your changes.




Et voila! Your citation appears in the text in just the same way as if you were doing it by hand, properly formatted with the correct page numbers.

This is an example sentence (Crone 45–7).

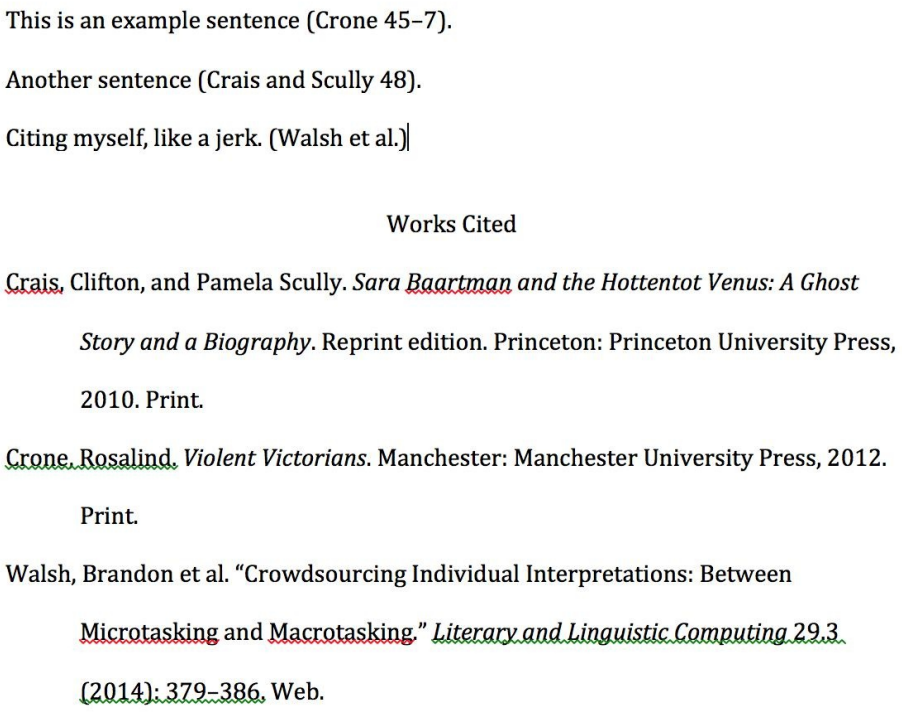
The process might appear a little slow, but once you get the hang of the workflow, this process greatly speeds up writing. Gather all your citations in one place, and then learn the handful of keyboard shortcuts for working with Zotero in Word. These are the ones we use most often:

- 'ctrl + option + a' will add a new citation.
- 'arrow keys' allow you to highlight particular objects from the Zotero search when inputting a citation
- 'return' selects a particular citation.
- 'cmd + down arrow' will bring up additional options like adding page numbers once you have a citation selected in the add citation input field.

Get the hang of these commands, and you'll save loads of time.

But adding citations is only one part of the process. You also want to add a bibliography to your document based on those citations. Zotero can do this too! From the Zotero menu in Word, select 'Add Bibliography.' Zotero will magically format the metadata you're using into bibliographical entries. Then format those into a bibliography based on the citation style you have chosen for the

document. By default, the bibliography will appear wherever your cursor was. So you'll need to move it around to put it in a location that works for you. Add a couple more citations using the other things we added to Zotero and see if you can get it looking reasonable. Here is what we came up with:



This is an example sentence (Crone 45–7).

Another sentence (Crais and Scully 48).

Citing myself, like a jerk. (Walsh et al.)

Works Cited

Crais, Clifton, and Pamela Scully. *Sara Baartman and the Hottentot Venus: A Ghost Story and a Biography*. Reprint edition. Princeton: Princeton University Press, 2010. Print.

Crone, Rosalind. *Violent Victorians*. Manchester: Manchester University Press, 2012. Print.

Walsh, Brandon et al. "Crowdsourcing Individual Interpretations: Between Microtasking and Macrotasking." *Literary and Linguistic Computing* 29.3 (2014): 379–386. Web.

To make this all look reasonable, we put the bibliography at the end of the text and hit return a few times after the last sentence to give it space (you often might insert a page break to put the bibliography on its own page). We gave it a centered heading. And we inserted a couple other citations to flesh things out.

We hope you see how powerful Zotero can be. Zotero saves a lot of time, and it can help organize your workflow neatly and naturally. Now, when you read something new that you think could be useful, in addition to taking notes you will also add it to Zotero. Later, when you go to write, that information can be easily added to your document without much of a hassle.

Zotero is much more that just a way to produce citations and bibliographies. It also allows you to share the fruits of your labor with others. The tool is free to use because its creators believe in the open and accessible sharing of knowledge. Even its underlying code is **open source**, meaning that it is free (and anyone can contribute to it). This open ethos of sharing pervades the use of the tool as well. Once you get a handle for putting together your own collections of resources, you can use Zotero for examining the collections of other people as well. Zotero allows people to organize themselves into groups and share metadata among all their members. These groups are frequently open for new members and visible by all, which means that, if you are a member of the right groups, you can easily find your way towards a variety of resources relevant to your interests. We've used Zotero ourselves for similar things, to find resources as we put together courses and lesson plans on digital humanities topics. We will not cover these aspects of Zotero in this lesson, but we encourange you to explore them on your own.

Zotero isn't perfect. Sometimes, an error might occur, and you might have to wrestle with your text a bit. But the payoff is worth it. Save your work often. Back things up. And use Zotero without fear.

# Further Resources

- Jason Puckett has a more advanced breakdown of Zotero in *Zotero: A Guide for Librarians,*

*Researchers and Educators*

# Exercises

# Exercises

1. Take all the readings you've done so far for this class and pull the sources into Zotero.
2. In a new document, practice using Zotero by adding citations for each of the sources to sample sentences.
3. Add a Zotero bibliography at the end of your document.
4. Register with Zotero to create an account. This will allow you to participate in groups, which you will do for your final projects.

# Cyborg Readers

# Cyborg Readers

# How Computers Read Texts

# How Computers Read Texts

If you have been dutifuly following along until now, it should be clear that computers and humans don't think the same way. With respect to text analysis, we can say that computers and humans have complementary skills. Computers are good at doing things that would take us a long time to do or that would be incredibly tedious. Computers can easily count and compare and will do so for pretty much as long as you tell them to do so. In contrast, humans are very good at understanding nuance and context. Thus, you wouldn't want a computer to do any close reading, or unpack the claims of a primary or secondary text; this is something you are far better at. By the same token, it's probably easier to have a computer list all the numbers between one and 45678987 than to do it yourself.

If such a disparity in skills exists between you and computers, you may be wondering why we're teaching a class on digital text analysis. Why bring technology into the equation when it is a poor approximation for a lot of the things that we do when we read? The answer is that there are a lot of instances where you can combine the nuance of human thinking with the quantitative power of computers to look at texts in new and creative ways. In particular, you can make computers do a lot of the repetitive work that you might find tedious.

To do so, though, you need to know a bit about how computers process texts. In many ways, they have a hard time understanding data. They can interact with and use information, but they make very few assumptions and even fewer interpretations about what they're working with. Any interpretative abilities that they do have been specifically programmed into the computer's software. So what follows, then, is a lesson in not taking anything for granted.

In the context of text analysis, all of this means that computers do not read with the same ease that we do. Consider the following sentence:

"We saw $8^{1/2}$."

Taken alone, the sentence doesn't tell us much. Its meaning depends a lot on the question to which we might be responding, and we can think of two possible questions with very different contexts:

> "How many movies did you see?
>
> "What movie did you see?"

In the first case, we might be responding with the number of movies that we had seen. It was a slow weekend, and we spent it at the local movie theatre hopping from film to film. It was a great time! In the second situation, we might be responding with the title of a specific film, *$8^{1/2}$* by Italian director Frederico Fellini. So one answer is a number, and one answer is a name. Since humans are good at grasping context, we would easily be able to distinguish between the two. In most situations, we would just adjust our understanding internally before moving on with the conversation.

Computers cannot make inferences like these, and this fact has serious implications: numbers and words have significantly different uses. Here are two further extensions of the conversation:

> If you add four to how many movies you saw, what is the result?

If we were talking about a number of movies, my response would clearly be, "Oh that's 12.5. Why are

you giving me a math quiz?" If we were talking about the Fellini film, we might respond, "What? Oh, we were talking about a title, not a number. We can't add things to a title." Again, humans have the ability to respond to context, infer, and adapt. Computers aren't nearly as flexible: they need to know ahead of time, in most cases, what kind of information they are dealing with. That way they can act as you anticipated.

Programmers have developed conventions for telling computers to distinguish between these different kinds of information, or **data types**. The distinction we outline above contains the two most important ones for our purposes here:

- **Strings**: characters, the stuff of words
- **Integers**: a whole numbers

The misunderstanding about films depends on a confusion around data types like these. If you go on to learn how to program, you might find slightly different names depending on the programming language, and you will be introduced to other data types as well. But the distinction between strings and integers is important for text analysis. You can perform arithmetic operations on integers while strings respond less well to such things. You can capitalize words, but not numbers. And computers generally want you to deal with similar objects: you can combine strings (words can become sentences) or add numbers, but trying to combine a string and an integer will break things.

But notice that our beginning scenario hinged on the ambiguity between strings and integers. How does a computer know whether we are talking about strings or about integers in cases where they could refer to either? How does it know that we want 8 to function as a word and not as a number in this context?

Programmers over the years have built a variety of functions and tools into different languages to get around some of these difficulties, but they still remain. When processing text by a computer, we have to account for such problems. We generally do this by following very strict guidelines for inputting information. This **syntax** works in much the same way as grammar does for humans - helping the computer to keep track of what we mean and what we want it to do.

In this case, we can tell the computer that something is a string or not by the presence or absence of quotation marks:

- 8 vs "8"

The computer looks at those quotation marks and can intuit the difference in datatypes:

- A number without quotation marks? That's an integer.
- Ah quotation marks. That means I'm looking at a string.

Programming and text analysis more generally are built on such subtle distinctions. A computer needs to have its hand held in order to recognize difference and similarity. To a computer, the following are entirely unrelated:

- 8 ≠ "8" ≠ "Eight" ≠ "Eighth"

The computer would not recognize the relationships among those four clearly related words. It goes even further: computers think of lowercase and capital letters as different characters entirely.

"H" ≠ "h"

These differences can be extremely frustrating when you are beginning to practice text analysis, but

don't worry: you don't have to reinvent the wheel. You benefit from years of programmers developing ways to account for these things. In any programming context, you probably have access to a range of utilities to capitalize things, lowercase them, convert integers to strings, convert date timestamps into words, etc. What this means is that sometime, years ago, someone first invented that wheel for you. A diligent programmer came along and told the computer that "h" and "H" have a special relationship along with how to navigate that link. You can benefit from their work.

But there are advantages to these rigid restrictions. By following them, we can get very detailed information about texts that we might otherwise gloss over. The first part of any text analysis project involves converting complex language into organized data that the computer can understand. This first step involves smoothing out problematic bits and filling in any gaps, all with an eye to the issues outlined above and in the chapter on "Data Cleaning."

"This is a sentence" ≠ "This" "is" "a" "sentence"

A computer would not recognize the two sides of the equals sign as being equivalent. The left side, after all, contains spaces, and the right side contains a series of smaller strings, since each word is in quotation marks. Annoying? Maybe. But also useful! After all, we are rarely interested in whole sentences. We commonly refer to individual words as **tokens**, and the process of breaking sentences into words then becomes called **tokenization**. This allows us to structure our text into a collection of pieces that we can manipulate more easily.

We can break things down even further once we've divided a text into individual words. While we often care about how many times each particular token or word occurs, we might also care about the different kinds of words. We might want to keep track, on the one hand, of all the different words in a text regardless of how often they occur. But we might also want a different kind of vocabulary list. Rather than counting all the words, we might just want to grab a single example of each token **type**. If we have the following document:

> Test test test sentence sentence

We have five tokens and two types ('test' and 'sentence'). A list of types might be good for getting a sense of the kinds of language used in a text, while a raw list of tokens could be useful for figuring out what kinds of words occur in which proportions. Depending on our research questions and interests, statistics like these can help us figure out what the document discusses as well as how it is being discussed.

If sentences are broken up into words, we might care also about breaking documents into sentences first. We have a name for that too: **segmentation**.

> "But wait," you say, "computers care about capitalization. So if we tokenize a text and try to compare 'word' and 'Word' they will think they are entirely different things!"

Good catch! You're right, those differences in capitalization often aren't meaningful. It is a fairly common practice to lowercase all the words after you tokenize them. This process is often called **normalizing** the data, since we are smoothing out inconsistencies that might get in the way of our research questions. This whole collection of processes of segmentation, tokenization, and normalization has a name of its own as well: **preprocessing**, all those things you do to data before you work with it. Depending on your interests, you might include other steps, such as tagging tokens for parts of speech or filtering out particular types of words.

Textual data is messy, and it requires a lot of work to convert it into usable material. Very often, the process involves even more steps than those that we outline here. But once you get a handle on the fixed set of methods for doing so, a whole world of possibility opens up. After all, the internet is *filled* with unstructured textual data, and we could learn a lot about our world by analyzing it. This field of

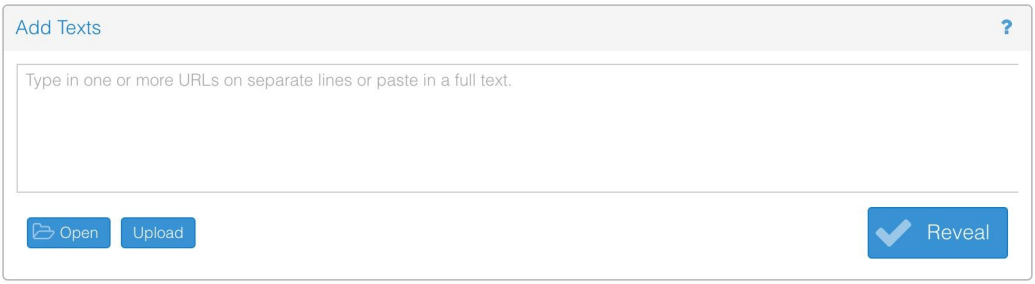
study is referred to as **natural language processing** ("natural language" refers to human languages like English, French, Arabic or Chinese, as opposed to computer languages, which were invented). A wide range of careers are built upon these foundations in fields in the sciences, medicine, government, and many more. The world needs people who can make sense of our textual world. You could be one of them.

# Voyant Part One

# Voyant Part One

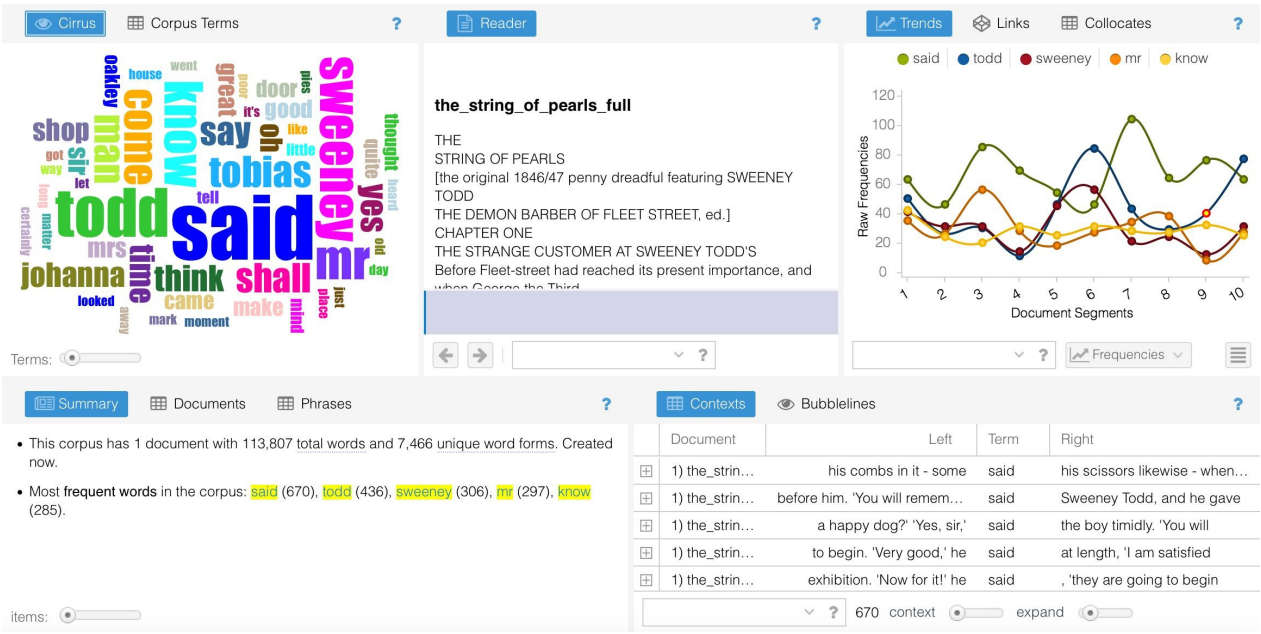
We will be using a tool called Voyant to introduce some basic topics in text analysis using cyborg readers.

Upon arriving at Voyant you will encounter a space where you can upload texts. For the following graphs, we have uploaded the full text of *The String of Pearls*, the 1846-1847 penny dreadful that featured Sweeney Todd, the demon barber of Fleet Street. Feel free to download that dataset and use it to produce the same results for following along, or upload your own texts using the window provided.




*Voyant Tools is a web-based reading and analysis environment for digital texts – please visit Hermeneuti.ca for more information..*

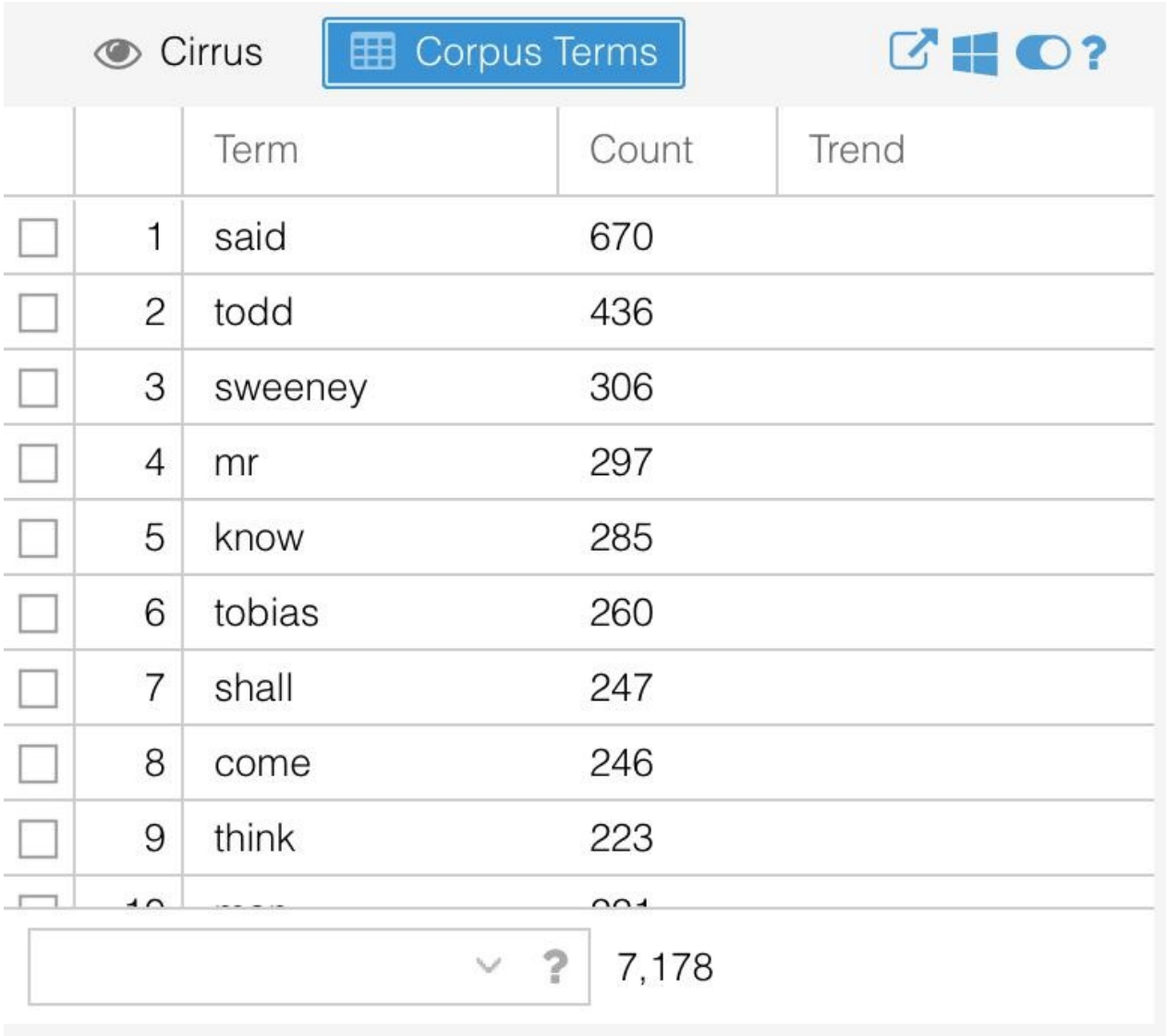
After Voyant processes your text you'll get a series of window panes with lots of information. Voyant packages several features into one tight digital package: each pane offers you different ways of interacting with the text.




Voyant gives you lots of options, so do not be overwhelmed. Voyant provides great documentation for working through their interface, and we will not rehearse them all again here. Instead, we will just focus on a few features. The top left pane may be the most familiar to you:


Word clouds like these have been made popular in recent years by Wordle. They do nothing more than count the different words in a text: the more frequent a particular word appears, the larger its presence in the word cloud. In fact, Voyant allows you to see the underlying frequencies that it is

using to generate the cloud if you click the "Corpus Terms" button above the word cloud.

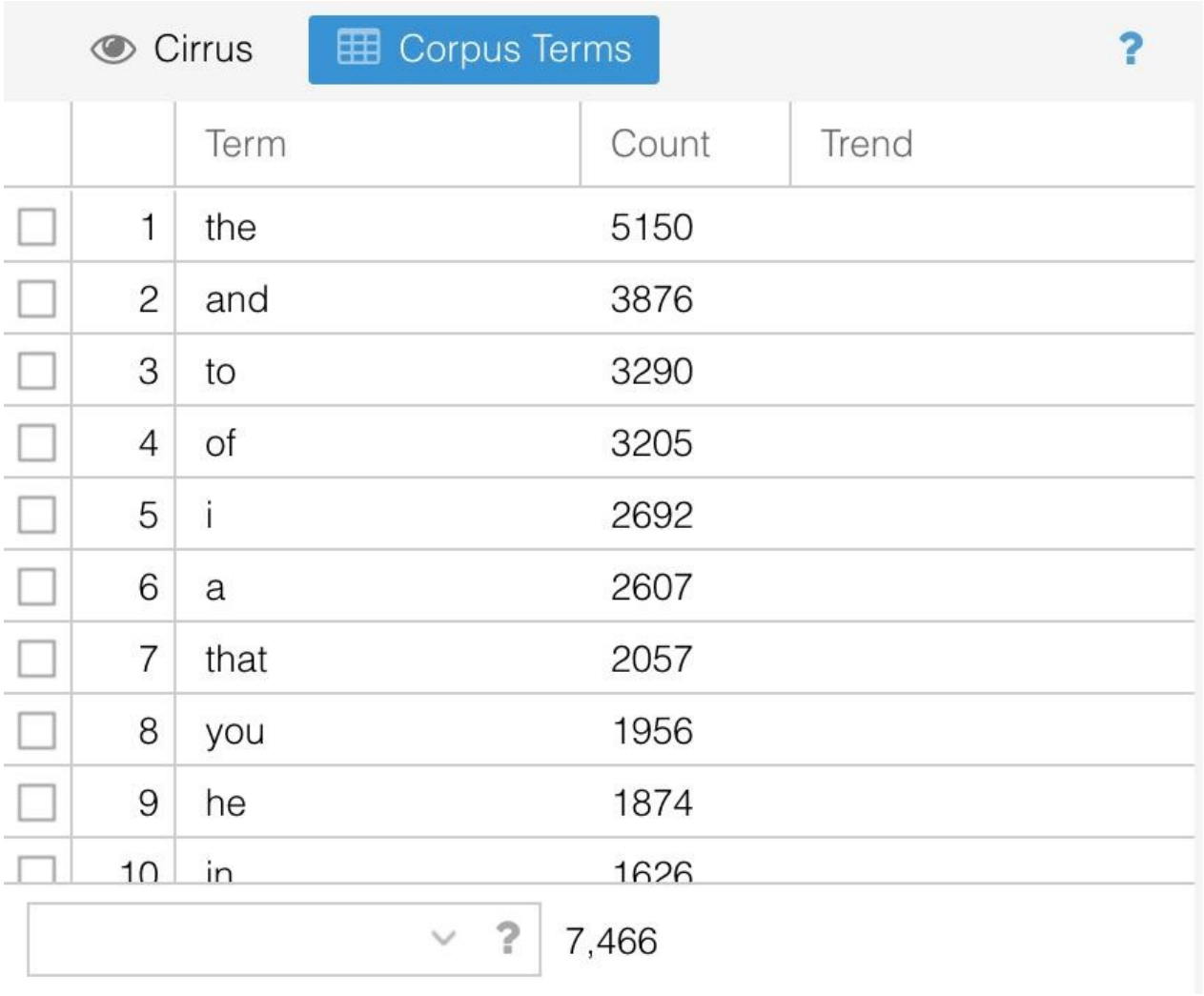



Concordances like these are some of the oldest forms of text analysis that we have, and computers are especially good at producing them. In fact, a project of this kind is frequently cited as one of the origin stories of digital humanities: Father Roberto Busa's massive concordance of the works of St. Thomas Aquinas, begun on punch cards in the 1940's and 1950's. It was one of the first works of its kind and was instrumental in expanding the kinds of things that we could use computers to do.

Busa's work took years. We can now carry out similar searches in seconds, and we can learn a lot by simply counting words. The most frequent words, by far, are 'said' and 'Todd," which makes a certain amount of sense. Many characters might speak and, when they do, they are probably talking about or to the central character, if they aren't Todd himself.

Notice the words that you do not see on this list: words like 'a' or 'the.' Words like these, what we call **stopwords**, are *so* common that they are frequently excluded from analyses entirely, the reasoning being that they become something like linguistic noise, overshadowing words that might be more meaningful to the document. To see the words that Voyant excludes by default, hover next to the question mark at the top of the pane and click the second option from the right.

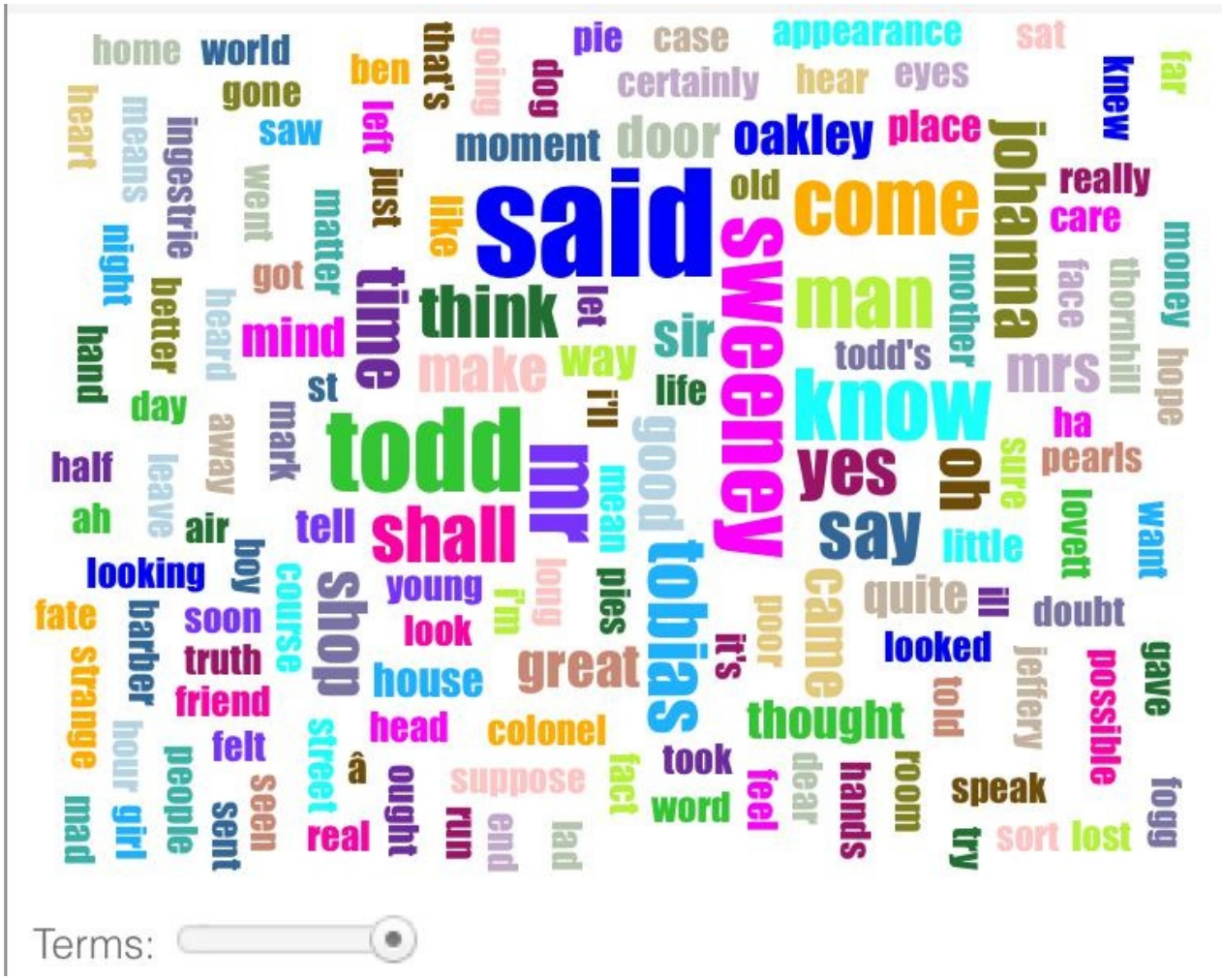
Use the dropdown list to switch from 'auto-detect' to none. Now the concordance will show you the actual word frequencies in the text. Notice that the frequency of 'said', the number one result in the original graph, does not even come close to the usage of articles, prepositions, and pronouns.




Words like these occur with such frequency that we often need to remove them entirely in order to get meaningful results. But the list of words that we might want to remove changes depending on the context. For example, language does not remain stable over time. Different decades and centuries have different linguistic patterns for which you might need to account. Shakespearean scholars might want to use an early modern stopword list provided by Stephen Wittek. You can use this same area of *Voyant* to edit the stoplist for this session of Voyant. Doing so will give you greater control over the tool and allow you to fine-tune it to your particular research questions.

There are some instances in which we might care a lot about just these noisy words. They can tell us *how* an author writes: those very words that might seem not to convey much about the content are the building blocks of an author's style. Tamper with someone's use of prepositions or pronouns and you will quickly change the nature of their voice.

Let's return to the word cloud. Using the slider below the word cloud, you can reduce or expand the number of terms visible in the visualization. Slide it all the way to the right to include the maximum number of words.




Just like the stopword list can be used to adjust the filters to give you meaningful results, this slider adjusts the visualization that you get. It should become clear as you play with both options that different filters and different visualizations can give you radically different readings. The results are far from objective: your own reading, the tool itself, and how you use it all shape the data as it comes to be known.

This is a good reminder that you should not think of digital tools as gateways to fixed and clear truths, either about historical periods or individual texts. Voyant may seem somehow objective in that it produces mathematical calculations and data visualizations, but now you've seen that you can easily alter the results. These techniques of "distant reading" are same as the models of "close reading" we talked about earlier in that both only lead to asking more questions and positing more interpretations.

That is a good thing.

## Interpreting Word Clouds

Given that what's important about word clouds is not producing visualizations so much as the interpreting the results, you might ask: what does this help us learn about *The String of Pearls*?

For one, looking at these word clouds suggests that much of the vocabulary of this novel either refers to or exists in the context of speech. One of the most prominent words is "said," but you also see "say" and "speak" and words like "yes," "I'll," and "oh" which probably -- although not necessarily -- come from written dialog.

Additionally, most of the words are short, one or two syllables long. Penny dreadfuls like *The String*

*of Pearls* were aimed at working class audiences: could the prevalence of these relatively 'simple' words reflect the audience of the text? In order to substantiate such a claim, we would probably want to look at other publications of the period to see whether or not this vocabulary was typical.

If we load Arthur Conan Doyle's "A Scandal in Bohemia" into Voyant, you can see that we get quite different results. (Again, feel free to follow along.)


A quick glance shows that the most common words tend to be longer than those in *A String of Pearls*. Indeed, the three syllable "photograph" is one of the most frequently used terms in this short story, one written for a middle-class as opposed to working-class audience. So maybe the simple vocabulary of the penny dreadful *is* related to the nature of its readership.

But let's not stop there! You may also notice that the word cloud for "A Scandal in Bohemia" has a lot of words related to high status: "king," "magesty," "gentleman," and "lady," for instance. In contrast, with the possible exception of the words "colonel" and "sir" in *A String of Pearls*, there are hardly any words in this novel that refer to rank. This gives you some indication that these two works are set in different social milieus in London.

Alternately, the types of words in these two works are not at all the same. The word cloud for *A String of Pearls* contains a lot of verbs ("shall," "said," "come," "know," "suppose," "thought"), whereas that for "A Scandal in Bohemia" is made up of a lot of nouns, particularly those referring to places ("room," "house," "street," "lodge," "window," and "adress"). This is an interesting thing to note, but you still want to think about what this means about the two different texts. Perhaps *A String of Pearls* is more concerned with action and on the excitement of people doing things than "A Scandal in Bohemia," where the emphasis is on moving through and exploring different spaces. Or maybe all of these observations are artifacts of the visualizations, things that the word clouds suggest but that might not actually hold up under scrutiny of the data. More thinking is needed!

Some of these conclusions were probably pretty obvious as you read these two works (or portions of them). You probably picked up the fact that *A String of Pearls* is set in working-class London, whereas "A Scandal in Bohemia" takes place in a more elevated milieu. You might even have noticed a difference in vocabulary, even if using Voyant made these distinctions more apparent and gave you further data to back up any claims you were making about them. But you probably didn't notice the emphasis on action vs. the importance of place in these two works. So this is a good example of how

reading with one eye to the computer can lead you to new interpretations.

## Further Resources

- Geoffrey Rockwell and Stéfan Sinclair, the creators of Voyant, have a great book on using it for text analysis: *Hermeneutica*.
- Shawn Graham, Ian Milligan, and Scott Weingart have an excellent introduction to working with humanities data in *Exploring Big Historical Data: The Historian's Macroscope*.

# Exercises

# Exercises

- How many tokens are in the following sentence? How many types?

'Speak!' cried Todd, 'speak! and speak the truth, or your last hour is come!

- Write out a normalized, tokenized version of the sentence.

Upload the text for *The String of Pearls* available here into *Voyant*. Analyze the results. If things seem particularly slow, you can try working with a smaller chunk of the text.

- Use Voyant to examine gender in the text. What kind of words do you need to look at? Which parts of Voyant? Make some sort of observations about your findings (3-5 sentences). Feel free to include a screenshot of the visualizations to help describe your observations.
- How would you measure moments of heightened suspense in the text? Take a spin at doing it if you think have a solid idea. Or simply theorize in 3-5 sentences.

Now upload the text for the various articles on *Lloyd's Weekly Newspaper about The Hampstead Murders* to Voyant and analyze them. This is the coverage of a late nineteenth-century murder case with a female victim and perpetrator.

- What is one other thing that you notice about the word cloud for this text? How might you back up these claims and interpretations if you were to read this series of articles? 3-5 sentences.

# Reading at Scale

# Reading at Scale

# Distant Reading

# Distant Reading

When Brandon was entering graduate school, an older student once summed up one of life's problems as a sort of equation:

> There is an infinite of material that one could read.
>
> There is a finite amount of time that you can spend reading.

The lesson was that there are limits to the amount of material that even the most voracious reader can take in. One's eyes can only move so quickly, one's mind only process so much. This might sound depressing, as if you're playing a losing game. But it can also be freeing: if you cannot read everything, why feel the need to try to do so? Instead, read what you can with care.

Computers flip the problem: their problem is not so much with quantity of reading as it is quality. As we have discussed before, computers cannot read with any particular nuance or understanding of what they are ingesting. Instead, technology might be best suited for helping us read at scale. Critics like Franco Moretti refer to this kind of analysis, when we use technology to get a bird's eye view of a corpus, as **distant reading**. If close reading, which we talked about earlier, gives careful attention to every word in a text, distant reading assumes that we can get new insight from thinking more broadly, by using computers to take in more texts than would otherwise be possible. Thus, we might have a computer give us schematic representations of thousands or even hundreds of thousands of texts. In the last chapter, we worked with stopwords and frequency analyses. We were mostly interested in the numbers of times that particular words appeared over the course of our corpus. Computers are especially good at reading for things just like this. On our own, we would never be able to read all 19th century British novels. But computers can help us to at least get *some* sense of this great body of work. Reading at such a great scale can also offer us a chance to chip away at what Margaret Cohen has called the "great unread", all that writing that has gone unnoticed because it never became part of the literary canon.

It might appear as though distant reading is less critical: after all, you could theoretically construct a beautiful program to analyze thousands of books for you without you having to ever crack open a single one of them. And some people do. As Matt Jockers, Ryan Cordell, and others have argued, however, even reading at this macro level requires attention to micro detail. Those same skills you were practicing with close reading earlier in the book? They are still deeply relevant. The work only begins once you have some results and a graph. You then have to figure out what elements are meaningful and what they might indicate. And the exploration very often takes you back to particular parts of the corpus that you want to read in more detail.

## Patterns

One way to begin thinking about developing approaches to using tools and methods like these is to take a step back. When looking at the results of distant reading, you are, more than anything else, looking for patterns and outliers. You could ask yourself a number of questions when looking at the results of tools like Voyant.

- Does anything clearly not belong or not make sense?
- What surprises you?

If you know your text is about the American South, and you find that the fourth most common token is 'France,' that probably says something interesting. You might need to revise your expectations and your research questions, and that is perfectly fine. This is actually part of the research process; if you don't revise your analysis, that means you aren't responding to what you are encountering in your sources. The most interesting thing about a project is rarely the first thing we think it will be.

- How do the numbers that the tool spits out at you connect with underlying concepts in the text?

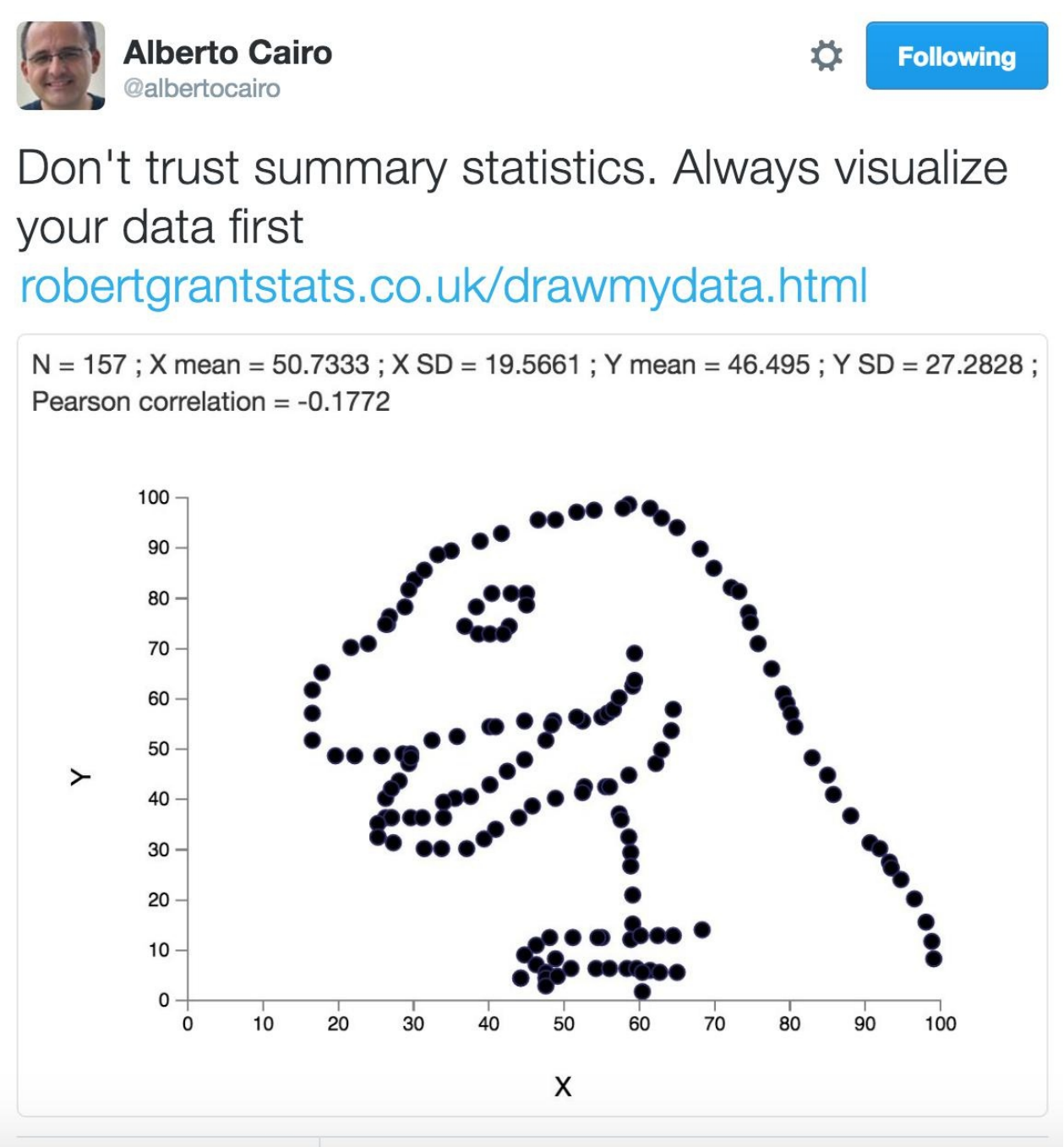
Our reading experience, our interpretations of a text, the way it makes us feel: these are the result of may things, but language plays a role in constructing all of them. Words form the basis for everything we get out of reading, so we can work backwards from word to concept. Think about what underlying concepts might be taking shape as a result of particular words. For example, if four of the top five words in a text are male names or male pronouns, that might say something about gender representation in the text. Personal pronouns might say something about what it means to be a self in your text. Four times more exclamation points than periods? That might say something about the rhetorical impression the author wants to convey.

- What trends do you see in the data?
- Is anything clearly decreasing or increasing over time?
- Are things largely the same over time?

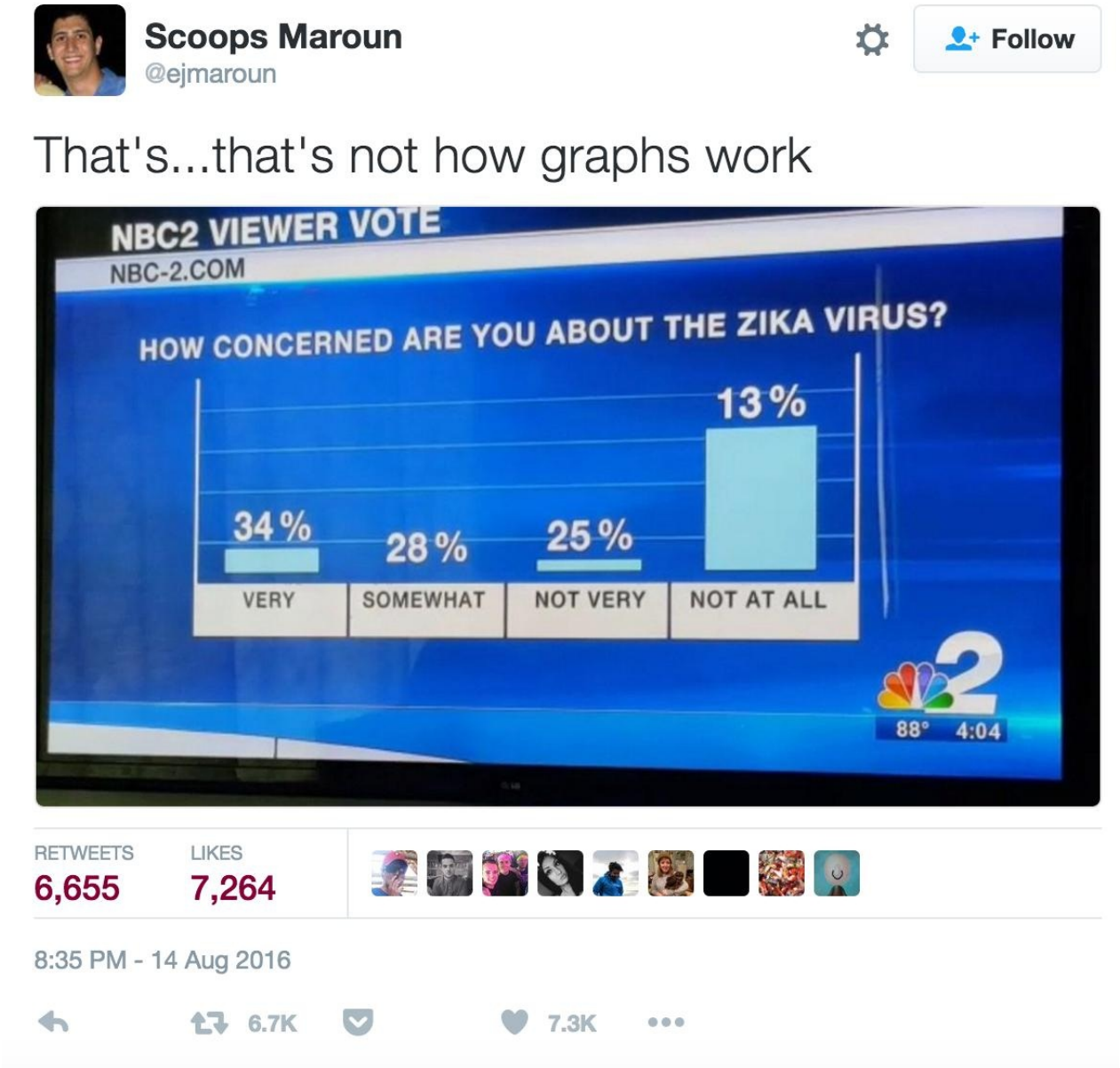
If you have a corpus where the dates for each text are known, you can begin to draw inferences based around language use over time. The Google NGram tool is built on such assumptions, though you should take care to think about how changes in language itself might affect your results (the classic example of this is the long s, which computers frequently read as an 'f' in older texts). The trends you see can offer good opportunities to reflect on your own understanding of what happens historically over the same time period. Alternatively, since we experience individual texts over time, we can examine how the use of a concept or word changes from the beginning of a text to the end. All of this might offer a way into thinking about the text as a whole.

- Does something just look plain wrong?

It is easy to think that the results the computer gives you are correct, and to take them at their word. After all, how could numbers lie? The truth is, however, that any data is the result of the biases of the people who produced them. Seemingly good statistics can make anything seem like objective truth when there might not be anything more than a pretty picture:




And a flashy visualization can just as easily show nothing.




Your own results might be the result of some setting that you have configured just slightly incorrectly. Or maybe you uploaded the wrong text. Or maybe you are misunderstanding how the tool works in the first place. If something has you scratching your head, take a step back and see if there is something wrong with your setup.

But wait, you say, I don't know enough about X to be able to do this kind of work!

You're fine! You don't need to know anything about statistics or computer science in order to be able to say something meaningful about texts through distant reading. Knowledge about both of these fields can go a long way and give you more meaningful and interesting things to say, but these tools, methods, and ideas should not be beyond anyone. Take a tool out for a spin and see what happens. You can always learn more about these fields to help give your analysis a stronger foundation, but it will all be for nothing if you don't even try because of such anxieties. Play first. Then enrich your work with further study.

You cannot read everything. Instead, focus on what humans are good at: reading with care and offering interpretations. The computer can work with big numbers much quicker than you. Your job is to help it do so in a meaningful way.

## Further Reading

- Ryan Cordell provides a helpful examination of the interconnectedness of close and distant reading in "Scale as Deformance"

In addition, the following resources offer great introductions to distant reading.:

- Franco Moretti, *Graphs, Maps, Trees: Abstract Models for Literary History*.
- Margaret Cohen, *The Sentimental Education of the Novel*.
- Matt Jockers, *Macroanalysis*.

# Voyant Part Two

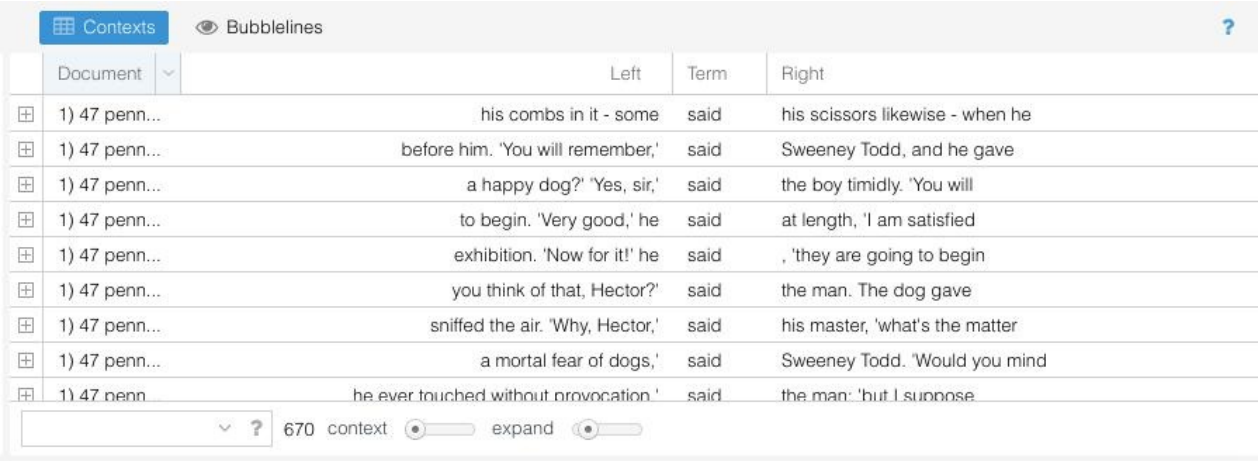
# Voyant Part Two

Let's look at Voyant in a bit more detail. Feel free to download the Sweeney Todd dataset and use it to produce the same results and follow along, or upload your own texts using the window provided. Look back at the word cloud that Voyant gave us for *The String of Pearls*:


Using the standard stopword filter in Voyant, the most common word by far is 'said.' Taken alone, that might not mean an awful lot to you. But it implies a range of conversations: people speaking to each other, people speaking about different things. One of the limitations of frequency-based measurements like these is that they only show you a very high-level view of the text. Once you find an interesting observation, such as 'said' being one of the most frequent words in the text, you might want to drill down more deeply to see particular uses of the word. Voyant can help you just do that by providing a number of context-driven tools.

In the bottom-right pane, Voyant provides a series of options for examining the contexts around a particular word. The first one is a **keyword in context (KWIC)** interface, Voyant's representation of one of the most common concordance tools. You can change the word being examined by selecting a new word from the 'Reader' pane. By adjusting the context slider, you can modify exactly how much context (i.e., how many words) you see around the instances of the word you are examining. Tools

like these can be helpful for interpreting the more quantitative results that the tool provides you. 670 instances of 'said' might not mean an awful lot, and the contexts pane can help you to understand how this word is being used. In this case, it can be useful for seeing different conversations: frequently, said followed by a name indicates dialogue from a particular character.




In this list of the first ten uses of 'said', two of them are closely joined with a name: 'Sweeney Todd.' If we look back at the word cloud for the text, we can see that these two words also occur with high frequency in the text itself. Given this information, we might become interested in a series of related questions:

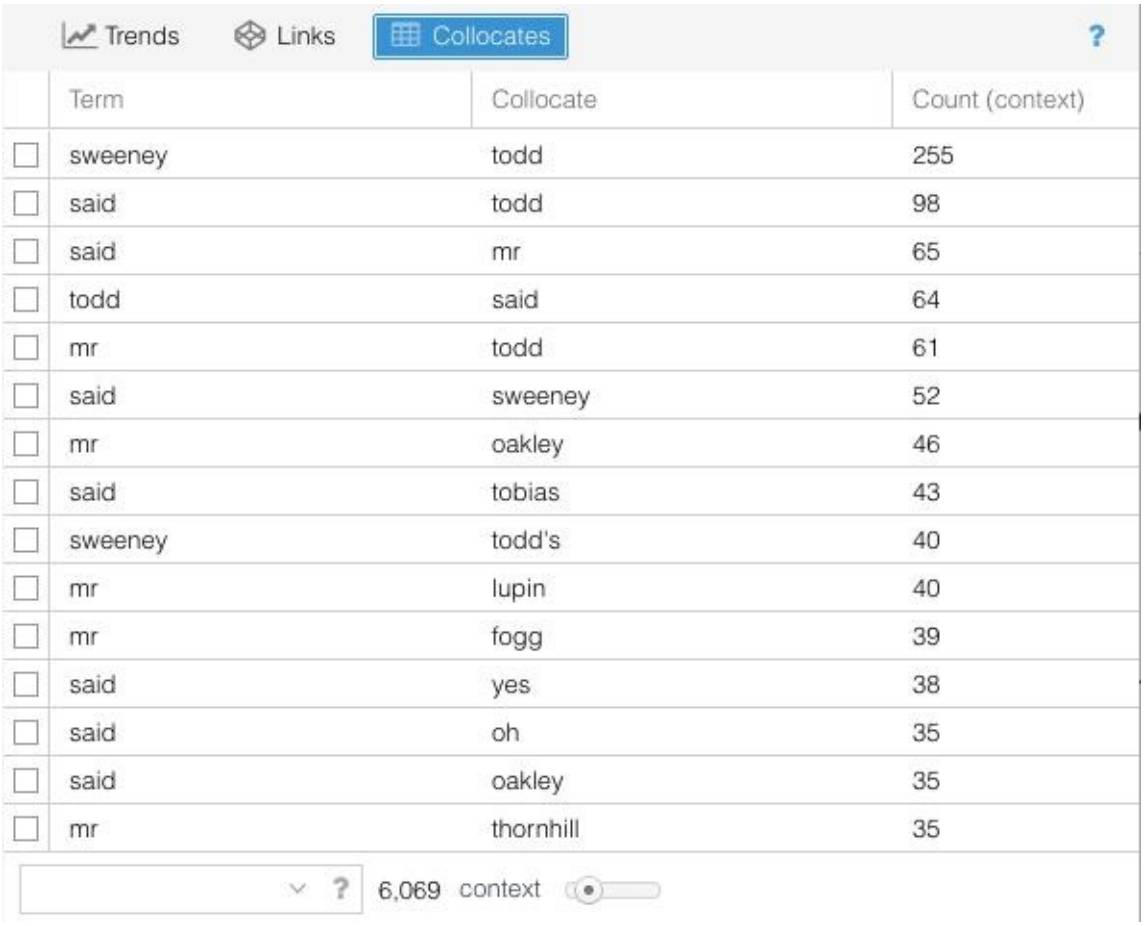
- How often is Sweeney Todd talking?
- What is he talking about?
- Who is he talking to?

As we move away from particular words towards clusters of phrases in contexts, we also need a new vocabulary to represent those relationships. You may have heard of **n-grams** from the Google Ngram Viewer, which allows users to search large corpora for specified words or phrases. An n-gram is a sequence of words that occurs next to each other of a particular length: the 'n' becomes a stand-in for the specified length of phrase. So take the following sentence:

"This is a sentence to illustrate ngrams."

"To illustrate" is an n-gram of length two, while "is a sentence" is an n-gram of length three. We use a convenient shorthand for referring to ngrams of this length: **bigrams** and **trigrams**.

**Collocations** are words that tend to occur together in meaningful patterns: so 'good night' is a collocation because it is part of a recognized combination of words whose meaning changes when put together. 'A night,' on the other hand, is not a collocation because the words do not form a new unit of meaning in the same way. We can think of collocations as bigrams that occur with such frequency that the combination itself is meaningful in some way.



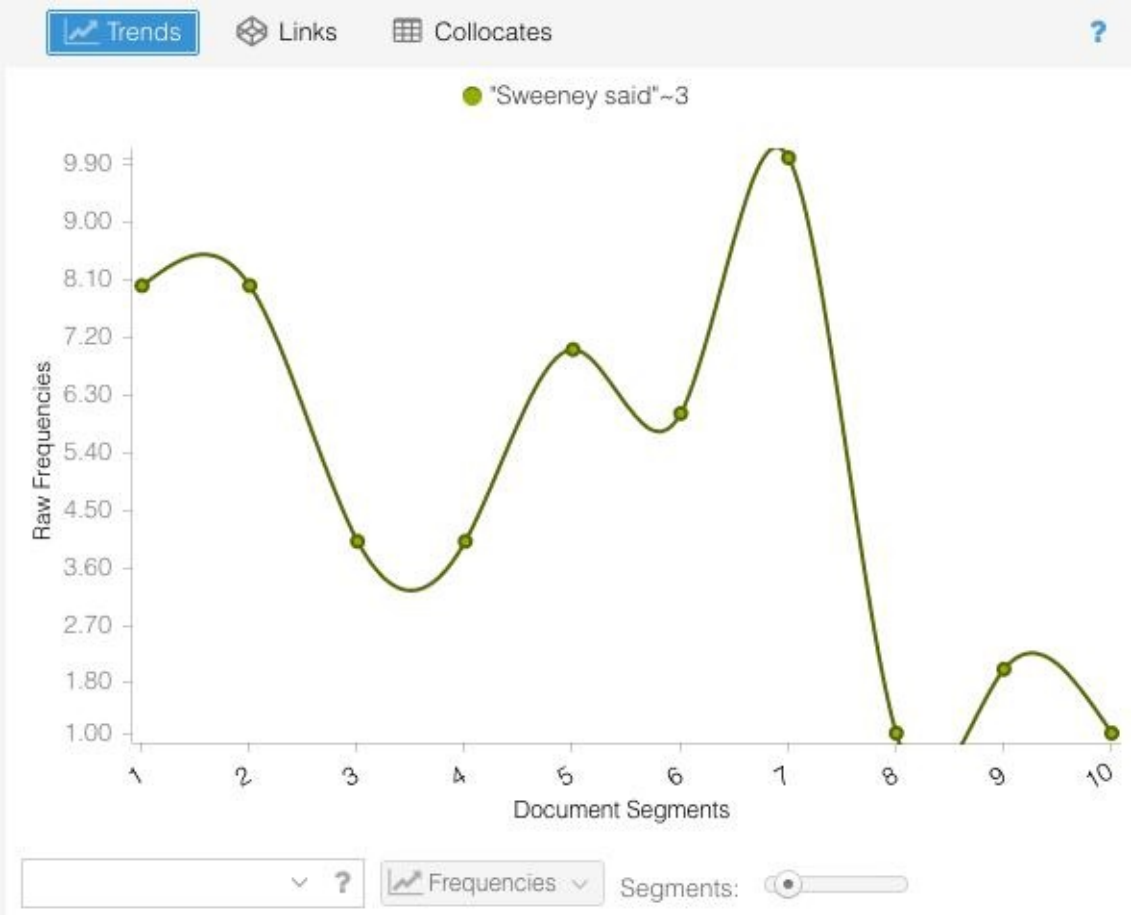

In a similar way, Voyant allows you to see the phrases and words that occur to next to each other in the text on a regular basis. The 'context' slider allows you to find sentences where two words occur near each other. So setting a context of three for 'sweeney' and 'todd' will give you all the three word phrases in which those two words occur: they do not need to be contiguous. In this case "Sweeney Todd said" would match, as would "Sweeney said Todd." Each row tells you how often those words appear within a certain distance from each other.

Looking at this data on collocations can lead to interpretive questions you might want to pursue. For one, a lot of the collocations are men's names (Sweeney Todd, Mr. Oakley, Mr Lupin, Mr. Fogg, etc.), whereas no female characters appear in the list of most frequent collocations. What might that tell you about the nature of the text and the way the author was shaping it to appeal to an intended male audience? Or, consider the very high number of collocations for "Sweeney Todd" -- more than twice the next highest collocation. At one level, this isn't terribly surprising. Sweeney Todd is the main character, after all. At another level, you might find it odd that his name was repeated so frequently in the text: how often did the readers need to be reminded of what his full name was? But maybe this gives you some insight into the nature of the work as a serial novel. If readers were consuming the work in weekly installments over the course of many months (while they were probably reading other serial novels), they may indeed have needed reminders about the main character's name (as well as other essential plot points). These are all potential avenues of interpretation; to make any of these arguments, you'd need to gather additional evidence -- but at least Voyant gives you places to start.

Click on the row that lists 'said sweeney 52'. Many of the windows in Voyant are interactive, and selecting something will modify the visualizations and options available to you elsewhere in the tool. Selecting a row here will modify a line graph that shares a space with the collocates table. You'll need to select 'Trends' at the top of the pane in order to see the line graph.

When you do, you will see a graph of the selected collocation over time. 'Sweeny' and 'said' occur

within a space of three words in highly variable amounts over the course of the text. By looking at the graph, we can get a rough idea of when Sweeny Todd speaks over the course of the narrative.




To graph things, Voyant breaks up your document into a segments (you can change how many it uses). Within each piece of the text it calculates how often the selected phrase or word appears. In this case, we might say Sweeney Todd talks significantly more in the first 70% of the text than he does in the last portion. Since you read the last few chapters of the novel, you might have a sense of why this is. The end of the text deals primarily with revelations about Todd's actions than his actions themselves. Of course, you wouldn't know this if you hadn't read portions of the text, a good example of how "distant reading" and regular, old-fashioned reading can and should enrich each other.

The 'trends' pane can be quite handy, and it will allow you to see individual words or phrases as they rise and fall over the course of a corpus. Think of it as the next step in critically analyzing a concordance. After all, texts occur in sequence, and we can learn a lot from examining the locations in which significant words tend to cluster. You can think of this graph feature as roughly charting the time of the narrative and as helping you think about the order in which words occur.

Thinking about language as it unfolds over time in this way can offer new opportunities for analysis. It can also raise issues. In a single text like *The String of Pearls*, composed over a relatively brief period of time, we don't need to worry too much about changes in language. But very often we might be studying loads of texts published over the course of many years, decades, or centuries. We cannot assume that language means the same thing over corpora like these. Words change over time. Your data and interests will determine how important this caveat is for you. Think carefully about whether any significant political, social, or technological events during that time could inflect how language works in the texts you care about. History can enrich your work or deeply complicate it.

# Exercises

# Exercises

Here are three projects that take distant reading approaches of various kinds:

1. Quantifying Kissinger
2. Viral Texts
3. Syuzhet Part One, Syuzhet Part Two

Select one of the projects and familiarize yourself with it. Answer the following questions:

- What is their object of study? What is their corpus?
- What research questions are they interested in answering?
- What methodologies do they use?
- Select one visualization from the project. Screenshot it, and explain what is going on in the image.
- What do you think about the project? What are some questions that you have about it? What interests you about it?

These projects may incorporate methodologies, tools, or programming languages that we have not covered in this book. Don't worry! You are not expected to understand everything.

# Topic Modeling

# Topic Modeling

# Bags of Words

# Bags of Words

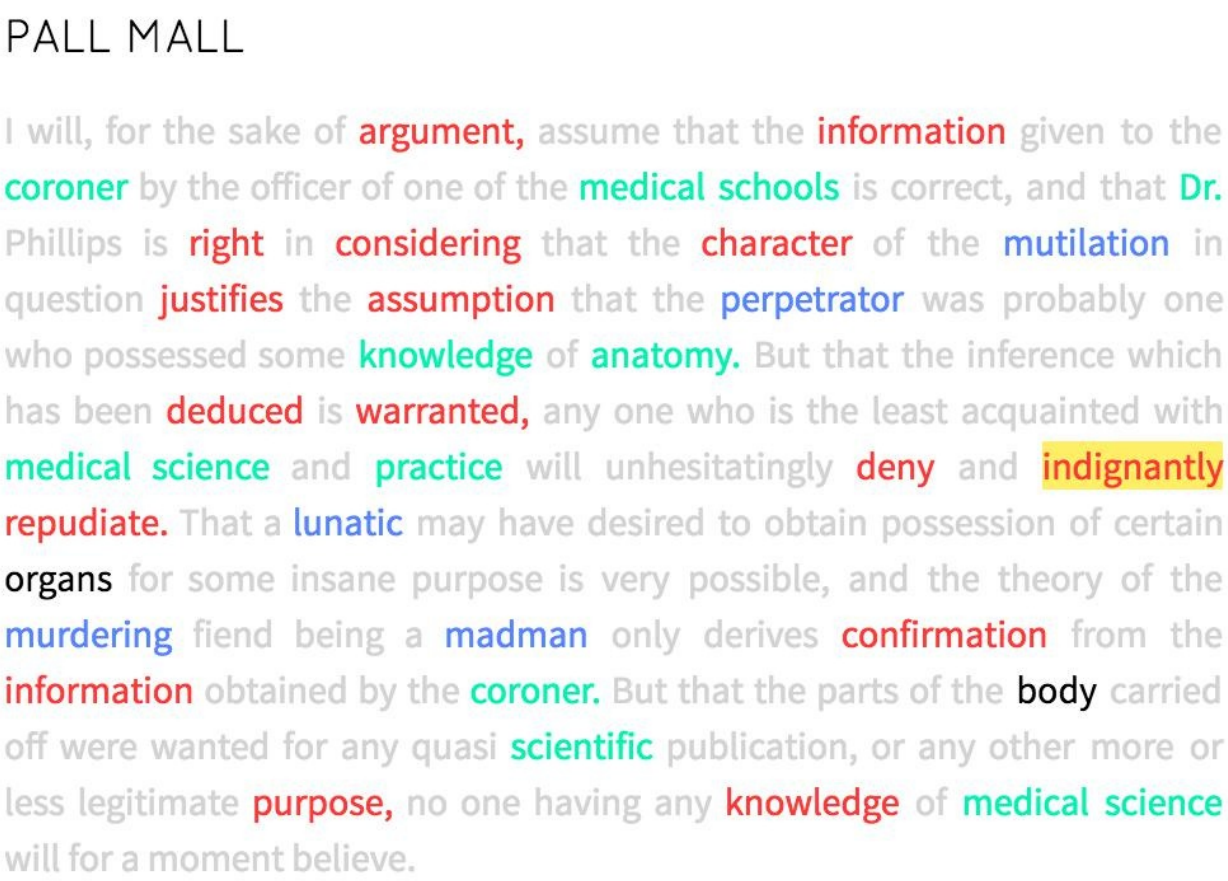
When we read, our eyes move in sequence across the page and take in phrase after phrase in the order in which they were intended. This fact allows us to do interesting things graphing words over time using Voyant. This sense of chronology is integral to how we, as human readers, understand texts. But it is possible to imagine other ways of reading. Have you ever skimmed over a page backwards looking at every other word? You probably still got the gist of the text even though you didn't read it in order and even though you missed many of the words.

Take this passage:

> I will, for the sake of argument, assume that the information given to the coroner by the officer of one of the medical schools is correct, and that Dr. Phillips is right in considering that the character of the mutilation in question justifies the assumption that the perpetrator was probably one who possessed some knowledge of anatomy. But that the inference which has been deduced is warranted, any one who is the least acquainted with medical science and practice will unhesitatingly deny and indignantly repudiate. That a lunatic may have desired to obtain possession of certain organs for some insane purpose is very possible, and the theory of the murdering fiend being a madman only derives confirmation from the information obtained by the coroner. But that the parts of the body carried off were wanted for any quasi scientific publication, or any other more or less legitimate purpose, no one having any knowledge of medical science will for a moment believe.

The excerpt is from a letter cited below about the Jack the Ripper murders from the *Pall Mall Gazette* published on September 28, 1888. Even without knowing anything about the context, you can probably infer a rough sense of the topic of the text: murder. We might further say that there are a number of overlapping topics in the text: evidence, medicine, murder, and many more. But how did you recognize these themes in the paragraph? If you skimmed the text, certain words might have lept out at you as indicating these topics. You see the words "coroner" and "body," and these words suggest particular things and not others. They make you think, "This article is about crime or medicine." They do not make you think, "Oh I'm reading a recipe for a nice guacamole" (or at least we really hope they don't). Vocabulary are the building blocks of the themes in a passage, and we can, theoretically, determine the topics at work in a text by paying close attention to the kinds of words that appear in it. Here is the same passage with one representation of how the reading process for this article might have taken place for you using *Prism*. We have highlighted various words associated with particular categories as such:

```
Highlight Color: Topic
---
red: evidence
green: medicine
blue: murder
black: words where two or more topics were marked the same amount.
grey: no topic marked
```




This is a visual model of how we might read the text. In this example, each of these colors represents a different kind of topic with which the text is dealing. Each topic is made of particular discourses: language associated with proving things, a vocabulary connected to medicine, and a series of words about crime. Other newspaper articles about Jack the Ripper could feature a different set of themes. The subject of the articles could change, the vocabulary would reflect this, and our sense of the underlying topics would shift accordingly. Probably most of these texts deal with crime, but you might have some Jack the Ripper articles that focus on the victims or on the police investigation into the murders.

Take a closer look at the kinds of words that we highlighted in the article above. Certain words might be really good indicators of a particular topic, while others might be fuzzier indicators of what we're talking about. For instance, although the word 'practice' might appear in conversations about both medicine and sports, a word like 'anatomy' is more closely and clearly indicative of a scientific topic.

Wouldn't it be interesting if we could somehow see the whole web of topics that occur in a text? And if we could find out how different topics appear in different texts and the degree to which the discuss them, we could figure out ways to distinguish between texts.

One problem with using *Prism* to do this work is that it depends on someone setting up these themes or topics beforehand, and thus in essence knowing or guessing something about the text before the text analysis process begins. But maybe there are some "hidden" topics in the text that we don't even know to look for. So what if we could use computers not just to distinguish between texts based on the topics they discuss, but also to find the very topics themselves?

We are beginning to float a different kind of reading. Let's take one more step back.

If we take the words in a text as being indicative of its underlying topics, we actually don't need to worry about word order so much. The sequence of words, sometimes called the **syntagmatic axis**,

only matters for certain kinds of reading. In previous chapters, we have preserved the sense of narrative time in a text - when we counted words with *Voyant*, we then graphed them over time. We cared about whether and how much a particular phrase occurred in the beginning of the text vs. the end. But we can find out interesting things about texts if we are a little more flexible and think about them not as things that unfold over time but rather as pure token counts, as **bags of words**. In a bag of words model, word order becomes irrelevant. All we care about is what words occur in a text and how often they do so. Pretty straight forward, right?

Take the following two sentences:

- "Fine. How are you doing?"
- "How are you doing? Fine?"

If we *normalize* a text by removing the stopwords, lowercasing the words, and getting rid of the punctuation, we get a bag of words. In this case, the bag of words for these two sentences is the same:

```
[
    "fine",
    "how",
    "are",
    "you",
    "doing"
]
```

The nuanced context of the sentences that makes the two of them different disappears, but we get the sense that they both discuss similar material. Now, we would not only want to know what words are being used; we'd also want to know how often they are mentioned. So a bag of words model for the following two sentences might produce something like the following:

- Sentence A: "Barbara is doing fine, thank you."
- Sentence B: "Thank you, Dave. I am doing fine."

```
Words in Corpus
[
    "Barbara",
    "is",
    "doing",
    "fine",
    "thank",
    "you",
    "Dave,
    "I",
    "am"
]

Counts for Sentences
A: [1, 1, 1, 1, 1, 1, 0, 0, 0]
B: [0, 0, 1, 1, 1, 1, 1, 1, 1]
```

Here we get two lists. "Words in Corpus" gives all of the words in our documents. "Counts for Sentence A" and "Counts for Sentence B" detail the number of times each of those terms occur in each sentence. So the first element of the Counts list for Sentence A is 1 because "Barbara" occurs 1 time. Sentence B has 0 in that same position because the word "Barbara" does not occur in the

sentence. We could have numbers as large as we need in order to represent the text as a whole. Pretty easy for a couple of short sentences, but imagine being able to break apart whole novels like this.

One last thing. Let's add this sentence to the bag of words model that we've been building:

- Sentence C: "I am Dave"

The new model looks like this:

```
Words in Corpus

[
 "Barbara",
 "is",
 "doing",
 "fine",
 "thank",
 "you",
 "Dave,
 "I",
 "am"
]

Counts for Sentences
A: [1, 1, 1, 1, 1, 1, 0, 0, 0]
B: [0, 0, 1, 1, 1, 1, 1, 1, 1]
C: [0, 0, 0, 0, 0, 0, 1, 1, 1]
```

Just by glancing at the counts for the three sentences, you could argue that two of the sentences are more similar to each other. Look at how many 1's you get in the sentences A and B vs. how many 0's you get in sentence C. You can do a lot of math to prove this, and even start to graph things to visualize the argument. Note that sentences 1 and 3 are mirror images of each other: they don't share any vocabulary in common. We can think about A and C as opposite ends of a continuum, then, and B being somewhere in between. Since Sentence B shares some with sentence A (both contain "doing," "fine," "thank," and "you"), but more with sentence C (both contain "I" "am" and "Dave," as well as no "Barbara" or "is" for either one), we can say that sentence B is a bit further to one than the other:

```
Sentences Graphed by Similarity
A------------------B----------C
```

For now, don't worry about the math behind all of this. We just want to give you a sense of the possibilities that can come from considering texts as bags of words. Note that, at a certain point, the vocabulary behind the model becomes irrelevant to this kind of thinking. We're just working with numbers, which is good for the computer! We can add the meaning and linguistic nuance back at the end, when we use this information to make humanities interpretations.

You might feel like this goes against everything that you've ever known about reading. This might feel like destroying a text. You're not wrong. This concept is pretty far removed from how we tend to read, since we tend to read in sequence across the page. This approach, instead, wants you to think about reading in a different way, to develop a new epistemology for the act. We lose something in the process, the sense of a text as it unfolds over time.

But we also gain the ability to think about a text in new ways. Sentences are just the beginning. You can use a bag of words approach to determine how different or similar whole books or authors are

from each other. If we have lists of words for each text as well as for the corpus (or set of documents) as a whole, we can actually work backwards to get a sense of the underlying topics that we were talking about a moment ago. Instead of skimming a paragraph to determine its basic topic, we could scan full texts -- and scan lots of them (Brandon's record is about 1.8 million texts in a corpus). And rather than trying to get a sense of 1-3 topics, we could break our text apart into 15-20 different topics. Now we are cooking with gas, and we're talking about topic modeling.

## Further Resources

- The Wikipedia page on the Bag of Words model was helpful for putting this lesson together.
- While we haven't quite gotten to topic modeling yet, Matt Jockers has a good summary description of how topic modeling and LDA work in these terms called "LDA Buffet: A Topic Modeling Fable."
- Daniel Chandler has a helpful explanation of the syntagmatic axis.
- David McClure's technical post on data mining the HathiTrust corpus with Python pointed us to Chandler's work.
- Ryder, Stephen P. (Ed.) "THE WHITECHAPEL MURDERSAN EMINENT MEDICAL MAN ON THE CORONER'S THEORY." Casebook: Jack the Ripper. http://www.casebook.org Accessed: 6 September 2016

# Topic Modeling Case Study

# Topic Modeling Case Study

In the previous section, we described how a single text could be broken up into a list of words and their frequencies, and we also suggested that a single text might be composed of any number of discourses or topics. Given enough time and energy, we can imagine a tool that would infer these topics for us without us having to read all of our documents first. The approach that we will take is a technique called **topic modeling**, a computational method that allows you to discover the topics that construct a text. Topic modeling does so by exercising a variety of statistical protocols over and over again on a text. Executing topic modeling projects yourself takes more hands-on programming than we want to introduce in this coursebook. So instead of exercising the techniques themselves or offering a tool for doing so, we are going to attempt to describe what topic modeling is and how to interpret results from it using a case study. Come talk to either of us in person if you want to go further and explore topic modeling on your own.

We will do a lot of statistical hand-waving in what follows, but we have some extra resources at the bottom of the page if you are interested in learning more about the mechanics of how topic modeling works. In the previous Bags of Words lesson, we walked through a paragraph and suggested that by skimming it you could infer the types of topics that it discussed. Topic modeling works in a similar way: the software looks for words that tend to occur next to each in statistically significant ways. The sum total of these words that occur next to each other becomes legible to a reader as a topic. The previous lesson looked at a single paragraph to think about the kinds of topics it might contain. You can imagine how this would get complicated very quickly when working beyond such a small scale. As humans, we have an upper limit to how much we can hold in our head: we can only keep a few topics in our brains at a time, a few texts in our thoughts. A computer doesn't have that same problem. Topic modeling software can process hundreds of thousands of texts, over and over again, refining its sense of how all the pieces fit together. It can give us a sense of the themes and discourses that run beneath an entire corpus.

So when you run topic modeling software, it looks for words that occur near each other in texts in meaningful ways over the course of the corpus. In most cases, it looks for words that occur in documents together. Remember, these words are not dependent on their location *within* the document. Topic modeling works on a bag of words model that only cares about whether or not the words occur within the text, not their position within it. But you might occassionally **chunk** larger documents into a series of paragraphs so that the software thinks about them each as separate documents for finer granularity. There are a number of similar tricks for refining your processes

After topic modeling a whole corpus, depending on the software you use, you will often be given a lot of information that can be hard to parse. All of these add up to the results of your project. The first important piece, a topic/term matrix, will contain lines that look something like this:

```
0 2.5 librivox httpupload mp orgshareupload duration mb het de heres

1 2.5 file volume audacity problem db files upload edit fix version (

2 2.5 text word read words dont make question change made pronunciat

5 2.5 recording project end librivox post files file number thread s(
```

```
7 2.5 english years life world language history people american scho

8 2.5 project page catalog httplibrivox found files link archive aud
```

The program spits out a series of "topics," which in this case are defined as sets of words that tend to occur in documents together with statistical significance. In these examples, we asked for the program to give us twenty topics; the identifying number of each topic is the first number on the left side. Note two things. First, the numbers start with 0 and end with 19, instead of starting with 1 and ending with 20. Second, we have selected a few topics to highlight out of the total number; not all twenty are listed here. After the numbers, you see the series of words that make up that topic. The computer does not know anything about these topics at all; it merely sees that these words often co-occur in documents.

The real work of topic modeling involves interpreting these topics in ways to make them meaningful to readers. Some of them are noise, artifacts of words that tend to occur next to each other. But others make more sense. Based on topics 1 and 5 you might be able to infer that these particular texts came from descriptions of sound recordings of some kind. Topic 2 might suggest that we are talking about reading texts. Much of the work of topic modeling involves trying to make assumptions about what kinds of discourses these clusters of words imply. In this case, the corpus under question is about 1.8 million forum posts from librivox.org, a site that produces public domain audiobooks. All of the examples in this lesson are drawn from work that Brandon has done examining these materials. Some more topics:

```
11 2.5 recording librivox post test find make start wanted record fo

12 2.5 noise sound recording good bit sounds hear mic voice backgrou

13 2.5 part read act line parts missing lines play great wrotei id e

14 2.5 section pl mw sections ready uploaded notes fixed corrected u

15 2.5 de la le je les pour pas en vous ne est si ce dans du des mer

17 2.5 ich die und der ist das du von es nicht den auch ein im noch

19 2.5 reading read great book listening books reader listen love re
```

Looking at these topics, we might see that a number of them deal with aspects of sound recording. In particular, the texts often tend to talk about the act of producing these recordings. This makes sense, as users of LibriVox actually create and upload recordings themselves. Topics 15 and 17 are also notable, as they represent French and German language topics. In any large corpus that is primarily one language, texts written in other languages will tend to group together. If we assumed before that medical language would tend to occur next to each other, this makes sense. French vocabulary is far more likely to appear next to other French vocabulary, in a text that is entirely French, relative to a larger corpus that is primarily in English.

For each document in the corpus, topic modeling also spits out a matrix showing what percentage each topic is likely to contribute to that text. The matrix might contain hundreds of thousands of segments that look something like this (modified slightly for readability):

```
0 httpsforum.librivox.orgviewtopic.phpf23t2
12 0.14788732394366197
1 0.13380281690140844
11 0.07746478873239436
```

```
2 0.06338028169014084
19 0.04929577464788732
18 0.035211267605633804
17 0.035211267605633804
16 0.035211267605633804
15 0.035211267605633804
14 0.035211267605633804
13 0.035211267605633804
10 0.035211267605633804
9 0.035211267605633804
8 0.035211267605633804
7 0.035211267605633804
6 0.035211267605633804
5 0.035211267605633804
4 0.035211267605633804
3 0.035211267605633804
0 0.035211267605633804
```

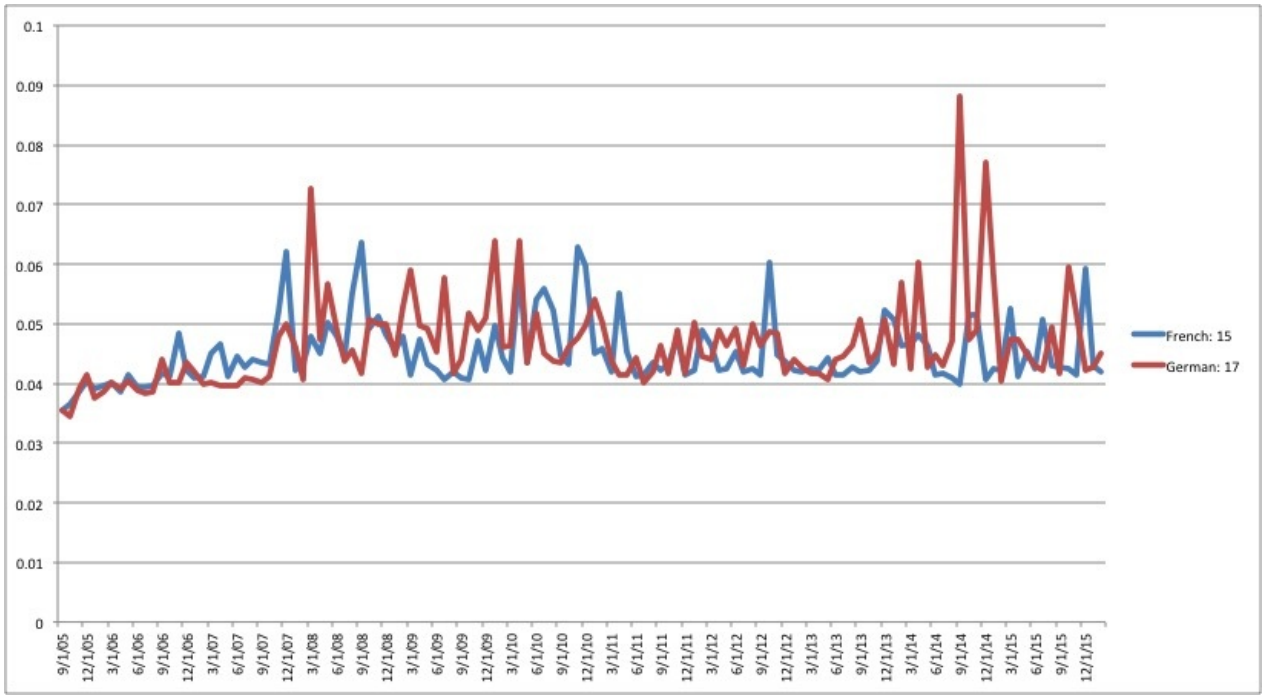
The first line here contains information about the document, an ID for the document and then a name for it. In this case, each forum post has its own URL, so we are using the URL for the ID:

```
0 httpsforum.librivox.orgviewtopic.phpf23t2
```

This post is post number 0, the first post chronicled by the topic modeling software. Brandon stripped out some of the punctuation, but you probably recognize what follows as looking sort of like a URL. This topic modeling information corresponds to a forum post that exists at this URL: https://forum.librivox.org/viewtopic.php?f=23&t=2. What follows is a list of each topic our topic modeling software produced and the weight of each topic for that document:

```
12 0.14788732394366197
1 0.13380281690140844
11 0.07746478873239436
17 0.035211267605633804
15 0.035211267605633804
```

So topics 12, 1, and 11, in that order, are the three most prominent topics in the document. Topic 12 has a weight of 14% in the document, and so on. Topic 12 is far more likely to appear in this document than topics 17 and 15. Using this information, if we know the dates that each text was published, we can actually determine the rise and fall in prominence of different topics over time (see the Further Resources on this page for cautions when doing so). We could see how discourse ebbs and flows over the course of our corpus. Remember topics 15 and 17, our German and French language topics? They aren't especially prominent in this first text, which makes sense because it was written in English. There is a pretty low likelihood that German or French words are going to show up in an otherwise English text, all things considered. But we can use this information to get a sense of where French and German texts are likely to occur by charting the ebb and flow of the French and German topics over time:




We get clear spikes in the German topic from September to December 2014, and slightly smaller peaks in the French topic in late 2007 and 2008. Since we know that these topics represent pretty coherent approximations of the use of French and German language in the corpus, we can use techniques like these to argue that these particular moments represent periods of heightened activity by the French and German communities of LibriVox users. Based on this evidence, we could make arguments about the global interconnectedness of the community of LibriVox users. We might then take this information back to the corpus and ask new questions based on this data.

- What kinds of books are important to these communities?
- What kinds of conversations are they having?
- Why do these languages spike at these particular times?

Until now, we have stressed approaching text analysis with a clear sense of your interests and the research questions that drive them. Topic modeling works a little differently: it is more useful for exploratory work. We call topic modeling **unsupervised classification** because we are asking the computer to analyze and mark a text without giving it any clear directions. We just say, "here is some text. Do your thing and tell me what you find." A **supervised classifier** would take information from us to help it make decisions. We might say, "read this text. If it has more than fifty uses of the word 'crime' mark it as 'detective fiction.' If it has fifty uses of the word 'love,' mark it as 'romance'" (more on supervised classifieres in the next chapter). Unsupervised classifiers like topic modeling instead know very little about the underlying texts that they are examining. Instead, they process them based on an underlying model.

In the last lesson we called the **bag of words** model an epistemology of texts, a way of understanding documents that might be different from what you were familiar with. In the case of the kind of topic modeling we have been discussing, that model could further be called **Latent Dirichlet Allocation (LDA)**. We won't go into any detail about the specifics of LDA, but it is important to know that this is the model you are working with and that LDA assumes that a text is constructed from a small number of topics.

Don't be alarmed if topic modeling seems much more abstract than the material we have covered until now. To really understand how topic modeling works under the hood, you will need to have a grasp of

a variety of different topics in machine learning and statistics. We are not so concerned that you understand these specifics. We care, instead, that you understand the idea behind it, have some sense of how to read make sense of other topic modeling projects, and be able to explain them to others in general terms.

## Further Resources

- Andrew Goldstone and Ted Underwood have a great case study of topic modeling *PMLA* that also includes lots of useful introductions to topic modeling.
- The Programming Historian has a good introduction for executing topic modeling yourself using Mallet. It gets technical, but the early surveys of what topic modeling is can be very helpful.
- For a more thorough explanation of how the algorithm behind topic modeling works, you might take a look at Lisa Rhody's class exercise for teaching LDA.
- Miriam Posner is helpful on understanding topic modeling results.
- Benjamin Schmidt in "Words Alone: Dismantling Topic Modeling in the Humanities" provides very useful cautions when working with topic models over time.

# Exercises

# Exercises

Read Robert Nelson's "Of Monsters, Men -- and Topic Modelling" from the *New York Times*. It's available here or, if you can't get behind the paywall, you can search for it on your favorite search engine; the link should carry you to the article. You can find more information about the larger project and individual topics here.

Answer the following questions about this project.

1. What kind of documents is Nelson working with?
2. What kind of assumptions or research questions does he have for these texts?
3. What general conclusions does he draw from topic modelling?
4. What conclusions could not be reached through close reading of these texts? In other words, what is the unique contribution of topic modelling to our understanding of the Civil War?
5. Look at the graphs/data for the other topics in the Richmond Dispatch. You can find the full list here. What do *you* notice?

# Classifiers

# Classifiers

# Supervised Classifiers

# Supervised Classifiers

In the lesson on our Topic Modeling Case Study, we talked about unsupervised classifiers. When topic modeling explores texts to find the underlying discourses at work within them, our texts were not really labeled in any way. We did not say, "topic modeler, go out and search for a topic called, 'medicine.' A medicine topic will consist primarily of the words 'anatomy,' 'science,' 'hospital,' etc. Let me know what else you find!" Instead, the topic modeling software came up with groups of words that it thought were related with relatively little input from us. This has the advantage of showing us patterns that we might not even know are there. Topic modeling is useful for exploring a corpus and discovering new things about it.

You could think of unsupervised classifiers as similar to a roomba. You hit a button, and the tiny little robot dutifully goes out and starts cleaning your floor. It knows when it reaches walls and corners that it should try to scoot around them. And its cleaning brushes are spinning furiously the whole time. You haven't told the machine how to clean, or how to navigate your floor. You just push the button and trust that it has inherent assumptions and protocols that it will follow. That covers the unsupervised part, but an unsupervised *classifier* is obviously more sophisticated and different in kind. Instead of cleaning your floor, topic modeling uses statistics to sort the words in your texts in such a way that you can get a sense of underlying patterns in word usage.

Let's try another example, adapted from Lisa Rhody's farmers' market game that teaches topic modeling. Imagine you have a bag with apples, oranges, and bananas in it. Imagine you also don't have any idea what an apple, an orange, or a banana is. Now we tell you to sort the things in the bag. You can probably still sort the fruit even without knowing anything about them, and you would do so by creating three piles. You take the first item, and place it into a pile on its own. Pull out a second item. Does it look similar to the first? No? Gets a new pile. Third item? Looks like the first one, so it goes next to that one. Each time you pull a new item, you compare it to all the others and revise your piles accordingly. At the end, you'll have three different piles, organized by fruit. But you didn't know anything about those fruits ahead of time. Topic modeling employs a few other variables in the process, so check out Rhody's lesson to learn more. For now, we will move on.

Now imagine, instead, that we give you a slightly different exercise. We give you a bag filled with dragon fruit, star fruit, and durian. Imagine that you don't know anything about these fruits. We say, "find me all the durian." You could sort the fruit into piles all day long, but, if you don't know anything about durian, you won't be able to pick out the fruit you need. So we give you a little **training** by first bringing in ten examples of durian for you to study. We say, "Look at them. Study them. Pay attention to these characteristics: durian have spikes, they are big, and they are yellow-ish." We might also give you several examples of non-durian fruit so that you can get a sense of what durian doesn't look like. This set of fruit, where we tell you the correct labels for the fruit, is called our **training set**. Now, you have something to work with! You pull a fruit. No spikes. So you start a pile called not durian. The next one has spikes. Durian! You keep going until you have two piles, one that contains fruit that you think is a durian and one that contains fruit that you think are not.

This kind of classification is called **supervised classification**. You had to be taught what the characteristics of a durian were before you could really do anything. We would call this collection of traits a **feature set**, and it might look something like this:

```
feature_set = {
```

```
'has_spikes': True,
'size': 'big',
'color': 'yellow-ish'
}
```

Don't worry too much about the brackets, equals sign, etc. These are just a common way of organizing the information so that the computer can read them. Here, we're just saying that this feature set defines what a durian looks like: the fruit has to have spikes, be large, and yellow-ish. This allows us to make a reasonable guess as to whether or not any one piece of fruit we pull out of the bag was a durian. Notice how you can only work in binaries: the fruit is either a durian or not. Your not-durian pile had star fruit and dragon fruit in it, since you weren't really able to distinguish between the two in this thought experiment. If we pulled out a star fruit, we could only answer something like the following:

```
fruit.is_durian?
>>> False
```

Or this if we were looking at a durian:

```
fruit.is_durian?
>>> True
```

The test is actually pretty simple in its results, even if the feature set that leads to them is more nuanced. True and False are referred to as **boolean data types** in programming, and these boolean values are used to test or represent whether something is just that - true or false.

We have been developing a series of tests for fruit types, but they might not be perfectly correct: after all, there are other fruits that are large, spikey and yellow-ish. A kiwano melon could have gotten thrown into the mix, and you might have incorrectly identified it as a durian. Or you might have gotten an unripe durian, which you incorrectly tossed in the wrong pile because it was green. So we could better characterize our two piles as "probably not durian" and "probably durian."

Likewise, maybe you want to figure out a classification system to sort bagels. So you ask: is it round? Yes. Then it's a bagel. Does it have black dots? Then it's a poppy-seed bagel. Does it have white dots? Then it's a sesame-seed bagel. Neither one? Mainly light brown in color? Then it's a plain bagel.

But wait: this dog fits all the criteria for a plain bagel, and it is definitely not a bagel. Our classifier can say, at best, "probably bagel" or "probably not bagel." And sometimes it's wrong. Sometimes life gives you a dog, and all you can see is a bagel. (Go here for more on this classification problem.)

The use of the word "probably" should be a clue here - we have drifted into probability and statistics. What we have developed above are very basic **naive Bayes classifiers**. Thomas Bayes was an eighteenth-century statistician, and this classifier relies on his underlying theory of statistics. There are other types of classifiers, but this kind assumes that each feature (size, color, spikiness in the fruit example; shape and dotted-ness in the bagel example) in our feature set will have some say in determining how to classify something that is unknown.

In a real-world situation, we probably would have given you negative examples as well, examples of fruit that are not durian so that you had a more nuanced sense of what you were studying. In the case of a naive Bayes classifier and our fruit example, the classifier takes the number of times that durian actually occurred in our training set as the **prior probability**. The classifier then combines this number with the actual features that we provided to give a weighted probability as to whether or not what it is looking at is a durian.

In this case, our labels are durian or not-durian, true or false, though you could have more than just

two labels. The classifier then picks the label with the highest likelihood. We have trained ourselves to classify fruit, and we could replicate that same process on durian at a later date. If a master fruit vendor comes along, she could probably tell us how accurate we were. We could then compare our accuracy to that of another person trained to classify fruit, and we could figure out who is the better classifier. We could even figure out the percentage of the time that each of our classification systems is likely to be correct!

This might all seem a bit removed from the kinds of work that we have been doing elsewhere in the book, but we wanted you to give a firm foundation in what classification is before we modeled an example relative to text analysis.

## Further Resources

- The NLTK book has a good section on naive Bayes classifiers. The book is a Python tutorial, though, so it will quickly get technical.
- A Visual Introduction to Machine Learning provides a very handy introduction to other types of classifiers.

# Classifying Texts

# Classifying Texts

At this point, you might be saying, "Supervised classification is all well and good, but how does this relate to text analysis? I'm going to go back to googling for animal photos."

Stop right there! We've got one for you. If you think you're tired, how do you think this dog feels? Impersonating bagels is exhausting.

Now that you're back and not going anywhere, we should acknowledge that your point is a good one. We wanted to stay relatively simple in the last lesson so that you could get a handle on the basics of supervised classification, but let's think about the ways you could apply this method to texts. The NLTK book (which you should check out if you want to go into more depth into text analysis) lists some common text classification tasks:

- Deciding whether an email is spam or not.
- Deciding what the topic of a news article is, from a fixed list of topic areas such as "sports," "technology," and "politics."
- Deciding whether a given occurrence of the word bank is used to refer to a river bank, a financial institution, the act of tilting to the side, or the act of depositing something in a financial institution.

Let's break each of these tasks down. Remember, a supervised classifier relies on labeled data for a training set. This sample data that you'll be using depends directly on the type of problem you are interested in. You could work backwards, and figure out what kind of training data you would need from the question you are interested in:

- To decide whether an email is spam, you will need lots of examples of junk email.
- To tag a news article as belonging to a particular category, you will need examples of articles from each of those categories.
- To determine the use of the word 'bank,' you will need examples of the word used in all these possible contexts.

In each case, it's not enough to just dump data into the classifier. You would also have to decide what feature sets you want to examine for the training sets for each task. Take the task of building a spam filter. To determine whether or not a text is spam, you would need to decide what features you find to be indicative of junk mail. And you have many options! Here are just a few:

- You might decide that word choice is indicative of spam. An email that reads "Buy now! Click this link to see an urgent message!" is probably junk. So you'd need to break up your representative spam texts into tokenized lists of vocabulary. From there you would work to give the classifier a sense of those words likely to result in unwanted messages.
- You might notice that all your spam notifications come from similar emails. You could train the classifier to identify certain email addresses, pull out those which have known spam addresses, and tag them as spam.

You could certainly come up with other approaches. In any case, you would need to think about a series of questions common to all text analysis projects:

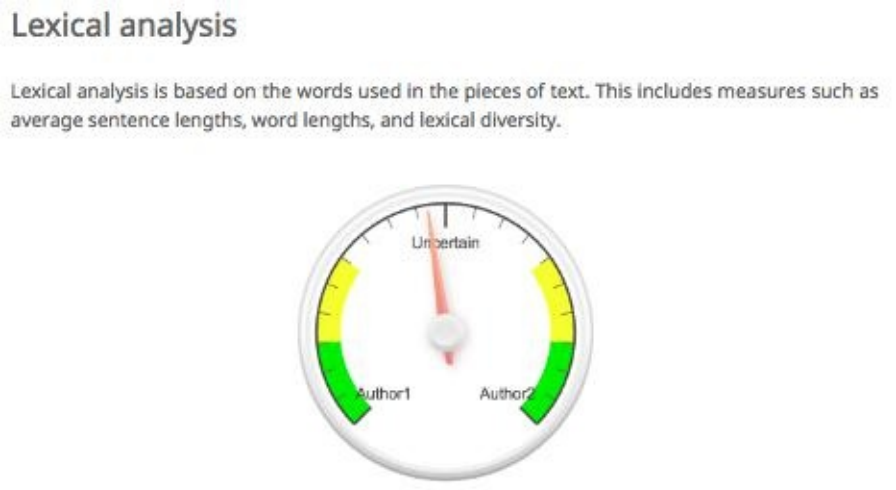
- What is my research question?
- How can my large question be broken down into smaller pieces?
- Which of those can be measured by the computer?
- What kinds of example data do I need for this problem? What kinds do I already have in my possession?

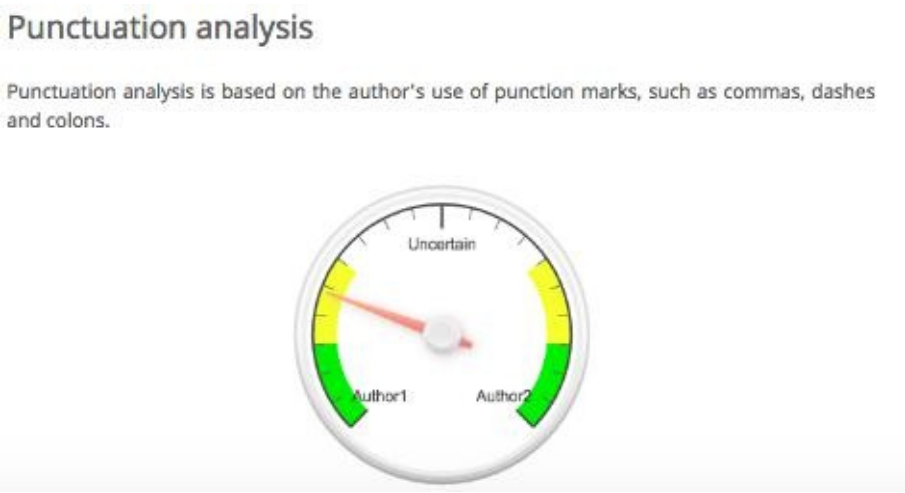
Going through these questions can be difficult at first, but, with practice, you will be able to separate feasible digital humanities questions from those that are impossible to answer. You will start to gain a sense of what could be measured and analyzed as well as figure out whether or not you might want to do so at all. You will also start to get a sense of what kind of projects are interesting and worth pursuing.

Now, let's practice on a supervised approach to a common problem in text analysis: authorship attribution. Sometimes texts come down to us with no authors at all attributed to them, and we might want to know who wrote them. Or maybe a single text might be written under a pseudonym, but you might have a good guess as to whom might be the author. You could approach this problem in a variety of unsupervised ways, graphing the similarity or difference of particular authors based on a number of algorithms. But if you have a pretty good guess as to whom the author of a particular text might be, you can take a supervised approach to the problem. To step through our same list of steps:

- What is my research question?
  - We want to be able to identify the unknown author of a text.
- How can my large question be broken down into smaller pieces?
  - We have a reasonable guess as to some possible authors, so we can use those as objects of study. We also are assuming that authorship can be associated with style.
- Which of those can the computer measure?
  - Well, style is the sum total of vocabulary, punctuation, and rhetorical patterns, among other things. Those can all be counted!
- What kind of example data do we have that we can for this problem?
  - We have the unknown text. And we also have several texts by my potential authors that we can compare against it.

To illustrate this experiment, we took two authors from our syllabus: Danielle Bowler and Pia Glenn. Using their author pages on Eyewitness News and xoJane, we gathered articles that belonged to each. Bowler tends to write shorter pieces than Glenn, so our training set included about double the number of pieces by Bowler (10) as by Glenn (5). With this body of training data for each author, we uploaded the texts to a great online authorship attribution tool by AICBT. The tool allows you to upload sample data for two authors. With this set, you can then upload a text by an unknown author, and the software will try to guess who wrote it based on a variety of text analysis protocols. In this case, the mystery text was "Freedom, Justice, and John Legend" by Bowler. Author 1 is Glenn, and Author 2 is Bowler. The tool attempted to identify the author of the mystery text as follows. The images also include AICBT's helpful explanation of the different metrics that they are using to analyze the unknown text.




This text is actually by author 2, so for the classifier to be accurate according to these measures, the arrow should point towards the right. You'll immediately see that we have some success, but also one failure! Function word analysis is a slight indicator of the correct author, lexical analysis is virtually useless, and punctuation analysis is way *wrong*. In a real classification project, we would want to use the success or failure of our classifier to revise our sense of which features are useful for our particular project. In this case, punctuation is not a good measure at all, so we would throw that out. Instead, we might focus on function words as an indicator of authorship. We can tweak our approach accordingly. But measures like these are only ever working probabilistically. We can say that the mystery text *might* be written by our second author, but only in rare cases could we ever know for certain.

Note also how these measures of authorship overlap with other things we have studied in this book. Remember stopwords, those words that are so common in a text that they are frequently removed before text analysis? In cases like this one, we actually care a lot about these simple terms. Two of the three measures for authorship here deal with just those words that we might otherwise throw away: punctuation, articles, etc. These words might not tell you much about the subject of a text, but they can tell you an awful lot about *how* a text discusses them.

Take a text that we've talked a lot about in this course: *The String of Pearls*. This penny dreadful was published in weekly installments and was written (we think) by James Malcolm Rymer and Thomas Peckett Prest. But the work was published anonymously, so we don't know which author wrote which chapter (or even if Rymer and Prest wrote the novel).

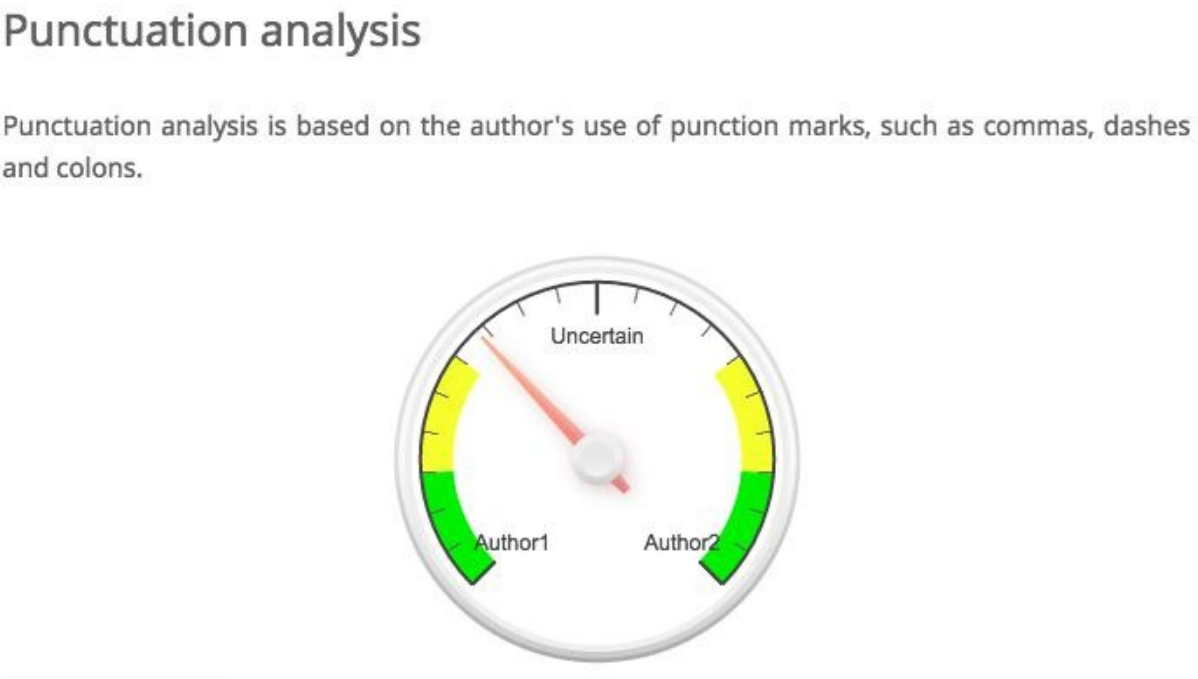
For the purposes of this demonstration, let's assume that we know that Rymer wrote Chapter One and Prest wrote Chapter Two. So who wrote Chapter Thirty-Three? If we go back to our author attribution tool and copy Chapter One into the box for Author 1 and Chapter Two into the box for Author 2, here's what we get for Chapter Thirty-Three:

## Function word analysis

Function words are content independent, and authors tend to use them in a consistent manner in their writing. Functions words for the samples above have been automatically identified: "the", "and"


## Lexical analysis

Lexical analysis is based on the words used in the pieces of text. This includes measures such as average sentence lengths, word lengths, and lexical diversity.




Here, it looks like the tool is trending towards Rymer as the author of the chapter, but we're mainly dealing with uncertainty. But that uncertainty itself is pretty interesting! Maybe what this is showing us is that the authors had a pretty similar style. Or maybe both authors had a hand in each chapter, and our training set is not particularly useful. If we had large bodies of text by each author we might have better luck. We might want to drill down further and investigate the uses of puncutation in different chapters or the lexical diversity, word length, and sentence length in much more detail.

- Are other penny dreadfuls similar to *The String of Pearls* in these respects?
- If so, what differentiates the style of these works from other types of serial novels?

Similar processes have been used for a variety of authorship attribution cases. The most famous one in recent times is probably that of Robert Galbraith, who came out with *The Cuckoo's Calling* in 2013. Using a similar process of measuring linguistic similarity, Patrick Juola was able to test a hypothesis that J.K. Rowling had released the detective novel under a pseudonym. You can read more about the process here.

If we can measure it, we can test it. And you would be surprised at just how many humanities-based research questions we can measure. Taking complicated human concepts like authorship and breaking them down into quantifiable pieces is part of the fun. It is also what makes the process intellectually interesting. If it were easy, it would be boring.

We have just barely scratched the surface of the field of **stylometry**, or the study of linguistic style using a variety of statistical metrics. You can carry this research process using a variety of programming languages, so you might take a look at our concluding chapter on Where to Go Next if you are interested in learning how to implement these sorts of experiments yourself.

# Exercises

# Exercises

Both W. T. Stead and Edward Tyas Cook were working for the *Pall Mall Gazette* at the time of the Jack the Ripper murders. Use AICBT's online authorship attribution tool to figure out which one was covering the case. Here is a link to the newspaper articles for you to test; be sure to find at least two that aren't just reproducing someone else's writing (like a letter to the editor).

Here is a link to W. T. Stead's works. You can find one of Cook's works here.

- What articles from the *Pall Mall Gazette* did you choose? Why did you choose them?
- What excerpts from Cook and Stead's writing did you choose? Why did you choose those excerpts? (Remember, giving the tool more data to train on means the more likely you are to get accurate data.)
- What did you find using the attribution tool? Are you coming up with clear results for either article?
- How confident are you that you can attribute either article to Cook or Stead? Why or why not?

# Sentiment Analysis

# Sentiment Analysis

# Sentiment Analysis

# Sentiment Analysis

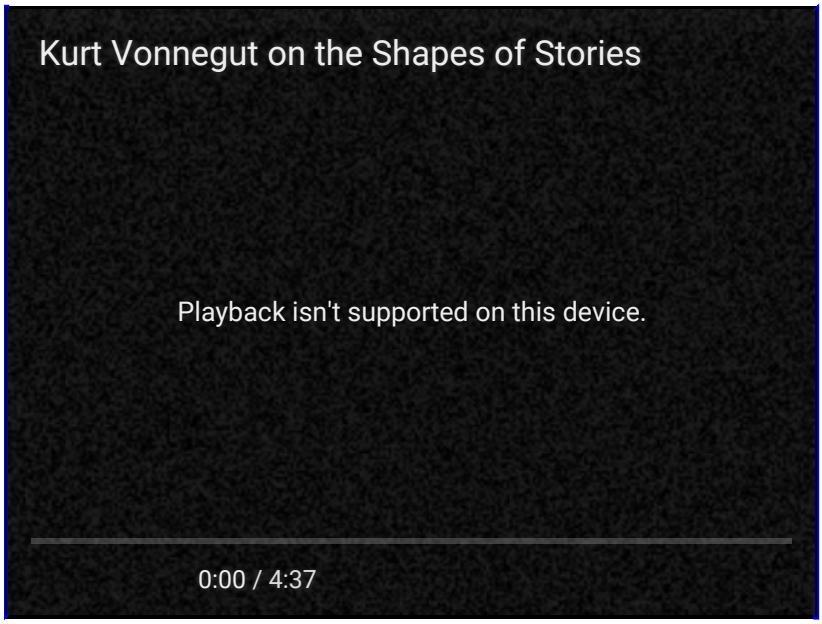
We began this book by talking about interpretation on a micro level: close reading asks you to pay attention to every small detail in a text to produce analysis. We have since zoomed out to think about what we could gain from macro reading and how computers enable us to understand texts in new ways. In our final moments, we will loop back around to the beginning.

We have repeatedly stressed the interplay of computation and interpretation: when the computer presents some results to you, your work has only begun. The computer can supply data, but you must interpret that data yourself. The computer does not really read. You do. What you've learned about is how to read *with* computers.

But you have probably also noticed in the last few chapters that the kinds of readings we are using our computers for have become more sophisticated. When we use software to discover the topics a text is discussing or to identify anonymous authors, we are not quite having them read in the same way as a person would. But we are getting closer. These techniques aim to provide a richer sense of a text, and they do so in quite sophisticated ways. We will close with a somewhat simpler problem, but one that is profoundly difficult for computers: is a particular text happy or sad? For that matter, is a sentence? A word?

This type of analysis that tries to capture the emotional resonance of a text is called **sentiment analysis**. You've probably engaged with this kind of work without realizing it. If you've ever been to Rotten Tomatoes to see what score a movie has gotten, you are looking at an aggregated number of reviews that have been marked as positive or negative. Businesses have a stake in such things as well. If you tweet about your recent flight, the airline would probably want to know whether you hated it or loved it. The former might result in you being directed to customer service, while the latter could result in a benign response like "thanks for flying with us!"

Sentiment analysis can also offer interesting opportunities for textual analysis. Check out this clip of a lecture by Kurt Vonnegut:



The idea makes enough sense as Vonnegut presents it: at certain times in a story, things are varying degrees of good or bad. As with any form of text analysis, this kind of information could be very useful for understanding a text.

- What kind of emotions does the author employ in the text? When?
- How do emotions map onto other aesthetic categories, like narrative structure?

It would be fascinating to have a computer that could easily mark the sentiments in texts for you. If you have been following dutifuly along, however, you should know that computers can't do much of anything without being explicitly told how. They can do very little in the way of understanding data without a human to guide them. Trying to extract complicated information like the sentimental arc of a text, how we are meant to feel about a sentence, or how an author intended us to feel are all complicated tasks that computers have a difficult time with. In fact, they can be hard for two different people to agree on. Try to guess whether these two sentences would be classified as good or bad:

- "I am very happy."
- "She is so sad."

Those were easy ones: good and bad. Hot and cold. How about this one:

- "It was the best of times, it was the worst of times…"

This sentence is from Charles Dickens's *Tale of Two Cities* and is probably a bit hard to parse in such a binary way. If it is both good and bad, it probably comes out as neutral, right? But Dickens was talking about the era of the French Revolution here; his whole point was that this was an extraordinary time, hardly a "neutral" situation. In fact, he is interested in juxtaposing different things - best/worst, London/Paris, etc. - not in resolving them. We would probably need some system for determining what to do in such situations.

Try this sentence, by Jane Austen, which complicates matters even further:

- "It is a truth universally acknowledged, that a single man in possession of a good fortune, must be in want of a wife."

An avid reader of Austen would know that her texts come loaded with satire. It is unlikely she actually means her words to be taken at face value. Virtually no truths actually are *universally* acknowledged to be true, so the sentence winks at the reader and should not be taken in full seriousness. In fact, much of her work is meant as a scathing criticism of the culture and people around her. These opinions are largely indirect, couched in irony and satire that asks the reader to read against what the text says on the surface.

All of these things are difficult to convey to readers, let alone computers. Sentiment analysis through technology is tricky, but that doesn't mean that researchers don't try. The process is difficult and riddled with error, but also intellectually interesting in a number of ways.

- How do we map complicated abstract ideas like emotion in a way that computers could understand them?
- What can sentiment analysis like this tell us about the objects that we study?

As with any form of text analysis, the potential uses range as widely as your imagination. One compelling recent use of sentiment analysis by David Robinson sought to gauge the degree of control that Donald Trump's campaign had over his Twitter account. Knowing that Trump tended to use a

Samsung Galaxy to tweet, Robinson wanted to determine if tweets from different techonologies might have different characteristics. If so, one could reasonably separate out his personal persona on Twitter from the one curated by his campaign stuff. Robinson found that we could reasonably determine that the angrier, more hyperbolic tweets came from a Samsung Galaxy (and were more likely to be by Trump himself). The tweets from iPhones were more likely to be "fairly benign declarations." With this knowledge we could reasonably trace the thumbprint of the Trump campaign handlers as distinct from Trump himself.

Computers might not be able to feel, but perhaps we can train them to know what emotions look like. The very idea of measuring sentiment computationally is provocative. If we were working in big business, we would care a lot about the results of such projects. But, as humanists, we can also gain a lot just from trying to model such complicated topics. The process is as enlightening as the product.

## Further Resources

- "The Universal Shapes of Stores, According to Kurt Vonnegut" has a brief explanation of Vonngeut's relationshop to the theory about plot trajectories.
- Maya Eilam has represented Vonnegut's theory about shapes in a variety of infographics.
- Jockers has a series of posts on his sentiment analysis project that begins here. These posts were where we first read about the connection to the Vonnegut clip.

# Sentiment Analysis in Action

# Sentiment Analysis in Action

To illustrate how sentiment analysis works, let's walk through a couple different projects. We will do a fair amount of handwaving at technical details, but hopefully you will get a sense of the kind of work that goes into sentiment analysis projects.

## Jockers and Syuzhet

Matt Jockers has been working on using sentiment analysis to discover plot trajectories in fiction in just the same terms as the video in the previous lesson (indeed, Jockers's writing is what pointed us to the Vonnegut clip in the first place!). By taking thousands of texts and classifying their sentences for sentiment, he has developed a software procedure for tracing plot trajectories and suggested that there are only six or seven different plot shapes based on this type of analysis. Jockers's bold claim has since come under serious critique by Joanna Swafford, who argues that the shapes are the results of configurations in Jockers's software rather than of any inherent quality in the text (also a recurrent theme throughout this book!).

Let's take a closer look at how Jockers is able to make such a claim. He uses a sophisticated software package that he constructed in the R programming language. We won't get into the details of the code itself, but we can cover the general approach. To find a more technical explanation you can look at Jockers's "Introduction to the Syuzhet Package."

Jockers's project combines supervised classifiers and unsupervised classifiers. Remember: supervised classifiers rely on training data that tells the software how to interpret and classify data. Unsupervised classifiers are not based on any prior training data. Instead, they rely on underlying assumptions and algorithms to categorize texts (in the case of topic modeling, this means that the unsupervised classifiers make assumptions about the relation between texts and statistics). We will focus on the supervised portion of his work below.

So, first, Jockers needed training data. In order for his software to read sentiment in sentences, it needed example sentences that had already been marked for emotions. By providing the software with example sentences, the software will be able to categorize related sentences in the future. So imagine that we train our computer with these sentences:

1. "I am happy!", positive
2. "I am sad!", negative

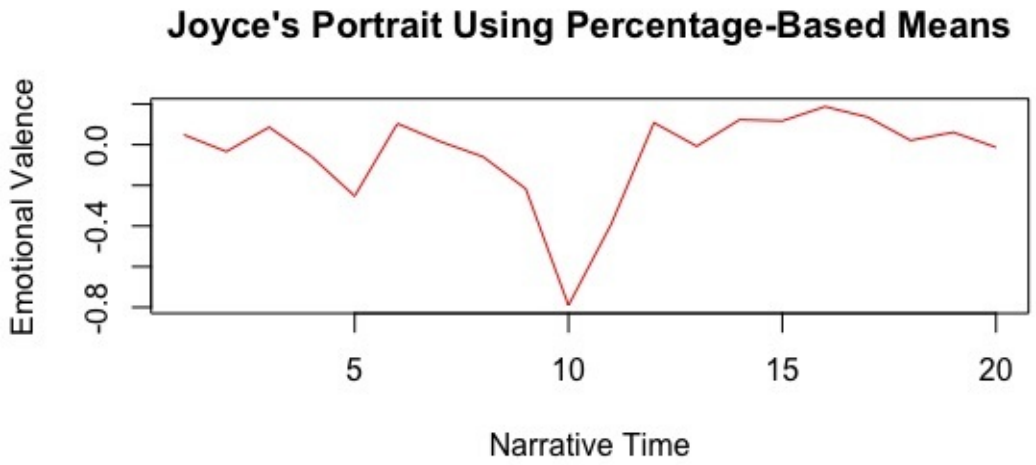
Later, we fire up our classifier and ask it to mark a given text for sentiment. Imagine that the computer encounters sentence 1 again later in the text. The computer could look in its bank of knowledge and remember that it should be marked as positive. But this classifier would not work very well: it only knows the sentiment for two specific sentences. When it encounters new sentences that we haven't pre-marked, it would not know what to do. In practice, we want to train a classifier on as much as data as possible to maximize its ability to handle new information. And we probably won't train it on full sentences. After all, computers distinguish between sentences and individual words in quite profound ways (we talked about this in "How Computers Read"). Depending on how thorough we want to be, we might give the computer vocabulary and phrases marked for sentiment instead. Since working with numbers gives us more options for graphing things, we might use "1" and "-1" to represent sentences with positive and negative values. And rather then a binary positive/negative, we

might mark for a continuum: numbers between -5 and 5, say. After all, 'good' is less positive than 'exuberant.' So each word or phrase gets converted into a series of positive and negative numbers.

You can find information on the training sets used by Jockers here. He uses a training lexicon of his own but gives the option to categorize sentiment using other training sets. Basically the software reads a text, looks at its memory of the training corpus to determine how positive or negative a sentence or word is, then converts the text into a series of values like this:

```
2.50 0.60 0.00 -0.25 0.00 0.00
```

Now the text is converted into a series of values that represent the sentiment of the text. As the numbers of the text becomes negative or positive, we get a sense of how the classifier reads the emotions of the text. From there, it is just a matter of plotting numbers to get a better representation of the sentiment trends over time. In the end, we can get something like this graph for James Joyce's *A Portrait of the Artist as a Young Man*, taken from Jockers's explanation of the software:

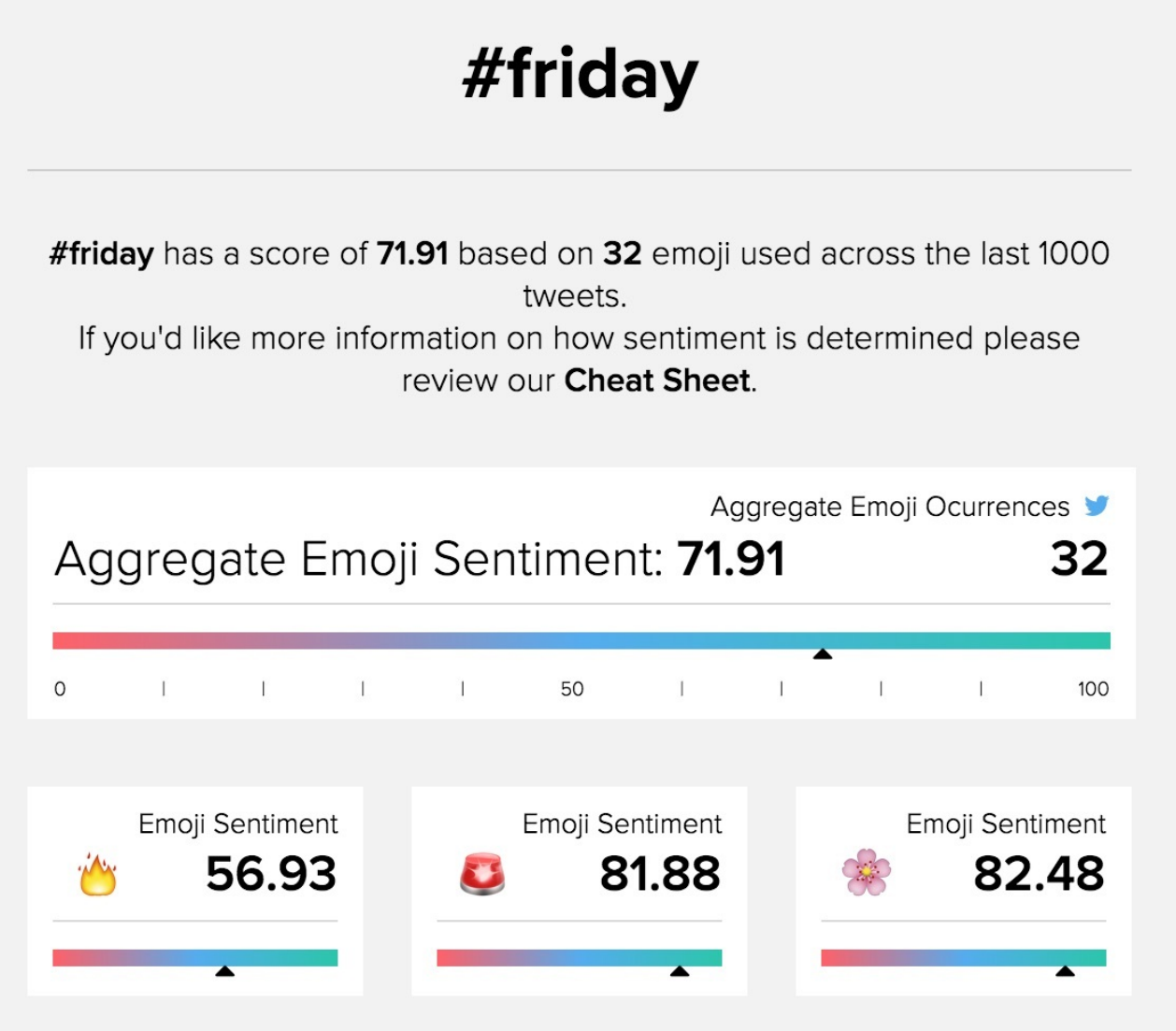



Just like we did with Voyant, Jockers breaks the novel into a number of segments and aggregates the sentiment for each section to get a sense of how the emotion changes over time. At its core, measuring sentiment computationally in this way relies on solid training data. The computer needs to learn how to map emotion-laden words and phrases onto some sort of numerical system. The robustness of your training set can strengthen or complicate your results. Getting a good training set can be difficult, however, since assembling one takes a great deal of time and labor. Notably, it is a lot of work to manually label single words with positive or negative valence. With a series of values like these for each text, Jockers then has a basis for comparison among his whole corpus. He can start to look for patterns in plot trajectories, which eventually leads to his claim that there are only a set number of plot arcs for novels.

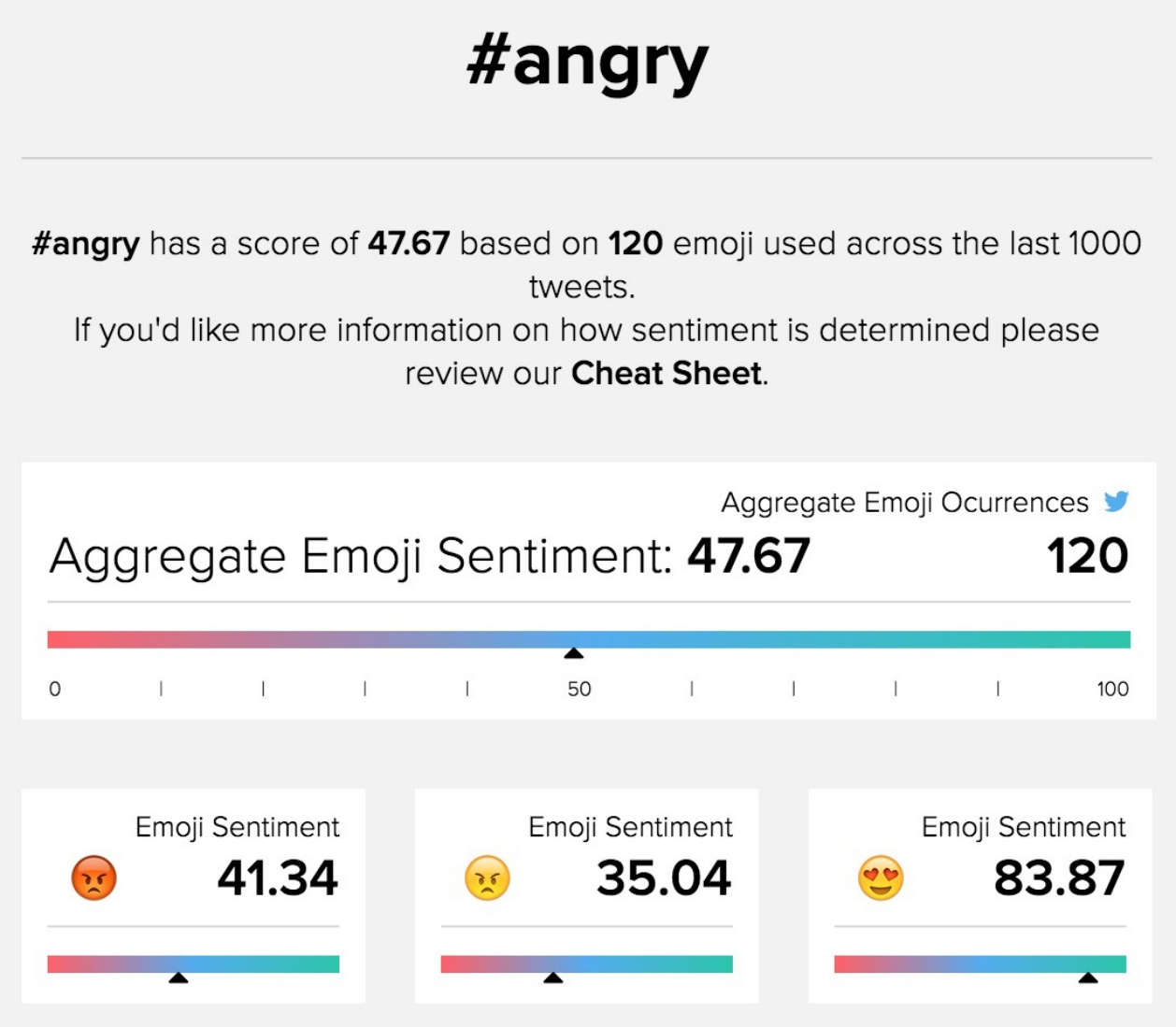
## EmojiSentiment

Another interesting use of sentiment analysis is EmojiSentiment. The project approaches the problem from a different angle: rather than trying to analyze textual content for sentiment, the site postulates that emojis embedded in tweets might be a good predictor of their sentiment. There are only around 2000 emoji and only a small subset have emotional valences to them; tagging these emojis is a lot easier than tagging all the words that convey emotion. The authors of this project use emojis to determine the overall sentiment for particular hashtag, as opposed to any one tweet. The project postulates that if you gather up all the emoji associated with a particular hashtag, you will get a pretty good sense of the emotional valence for that stream of conversation. For example, EmojiSentiment

reads #friday as being relatively positive:




On first blush, this makes sense. Everyone gets excited for the weekend. But look more closely. The three most dominant emoji are fire, police sirens, and flowers, which might seem a bit unusual. Note that we're only getting 32 emoji in the last 1000 tweets - these might just be artifacts of whatever is going right now, and the project can't handle more volume than that (it's a student project. Yay students!). So our sample size is rather small and could easily be skewed by a handful of active users. We would want far more emoji to really get a good measurement of sentiment. If #friday got a relatively happy average sentiment, #angry is much more negative:




Note how even #angry just barely dips below 50 to be predominantly negative. We actually had to search around for a while to find a hashtag that read as predominantly negative. Could this be a function of language - are people just happier than they are sad? Certainly not. This might have to do with how we represent our emotions on social media; maybe we don't use emojis to represent negative emotions that often. Or maybe sentiment analysis by way of computation is imprecise at best. To improve on this project, we would want to scale it up to read vastly more data. We might then use the emoji a tad differently. Instead of using them to measure sentiment, you could use them to train the sentiment classifer further so that it refines itself over time! All of this would require far more funding than the EmojiSentiment team has, however, and the tool is a great provocation as it stands.

- What might you imagine using sentiment analysis for?
- What kinds of texts lend themselves especially well to reading for emotion?

# Exercises

# Exercises

- Take the following text, and mark it on our class Prism (The Prism has a longer excerpt, not just what you see below):

  > Anita Sarkeesian doesn’t give me the address of her San Francisco apartment over email. Instead, she texts it to me a few hours before we’re set to meet. After thousands of rape and death threats, a bomb scare and an email promising a mass shooting at one of her speaking events, a woman can’t be too careful. For some male gaming aficionados, the most frightening enemy isn’t an animated foe but this 31-year-old feminist with a penchant for hoop earrings, sitting across from me. They’ve called Sarkeesian a con artist, and raised thousands of dollars to film an exposé-style documentary about her (which exposes nothing). Some even created a game in which users can punch an image of her face until it is bloodied.

- Now take the same text and process it on the live demo for Stanford's sentiment analysis software. (Note how each sentence is its own tree. The demo color codes individual words as very negative, negative, neutral, positive or very positive).

- What are the differences between how the emotions are marked in the Prism versus how they are marked in Stanford's tool?

- What does all this say about what humans are better at? Computers?

# Conclusion

# Conclusion

# Where to Go Next

# Where to go Next

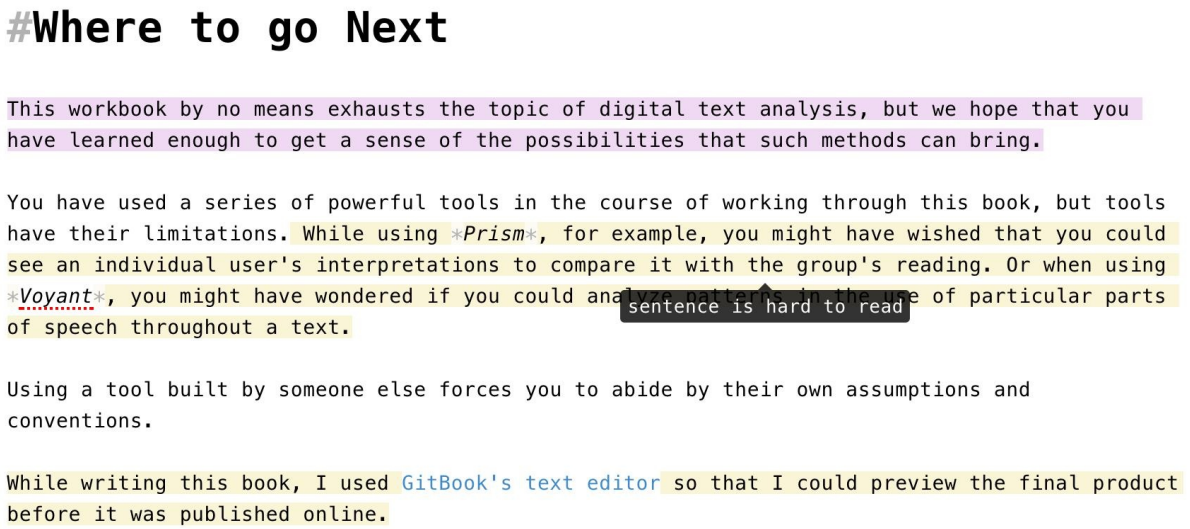
You've come a long way! Thanks for sticking with us!

You've learned a lot. Thought a lot. Read a lot. What comes next? We've talked about a bunch of different approaches to text analysis, so hopefully you have some ideas about things that interest you. One way forward might be to experiment with new tools. Or you could delve more deeply into a particular approach that piqued your interest. Perhaps while moving along you found yourself wishing that a tool could do something that it just wasn't set up to do.

- "If only it would do X!"

Maybe you can make something for yourself to do X!

While writing this book, we used GitBook's text editor so that we could preview the final product of our writing before it was published online. But the text editor has all sorts of added features. For example, it offers a style editor that suggests certain kinds of sentences might contain difficult syntaxs or formulations. Potentially useful, right? Maybe, but we turned it off right away. We found it really annoying to type while our text was screaming at us like this:




The most irritating thing was that we could not tell what metrics they were using to diagnose our writing. What makes a sentence difficult? The number of words in each sentence? The number of clauses? Subjects in particular positions? We have all sorts of opinions about why writing might be unclear, but, as best we could tell, the editor was mostly basing their suggestions on the number of words in each sentence. We turned the feature off and went on with our lives, but not before noting a truism of working digital humanities: using a tool built by someone else forces you to abide by their own assumptions and conventions. We could not change how the style checker worked beyond the minimal configuration options given to us in the platform.

You might have had similar feelings while reading this book. You have used a series of powerful tools

in the course of working through this book, but each one has its limitations. While using Prism, for example, you might have wished that you could see an individual user's interpretations to compare it with the group's reading. Or when using Voyant, you might have wondered if you could analyze patterns in the use of particular parts of speech throughout a text. Or maybe you were really interested in sentiment analysis or topic modeling. We didn't really offer tools for either of these approaches, because they quickly get a little more technical than we wanted. You need to be comfortable with some basic programming approaches to use tools of that nature.

A logical next step for you might be to learn a programming language that can help facilitate textual analysis. Python and R are two widely used languages for these applications with a wealth of resources to help get you started. Exploring a new programming language takes time and dedication, but it can help guide you towards new types of questions that you might not otherwise be able to ask. Learning to program can help you determine what types of questions and projects are doable, and which ones might need to be let go. Most importantly, it can help you realize when using a tool someone else has built is better and easier than reinventing the wheel. While we would have loved nothing more than to turn you all into self-sufficient Python gurus, we believed that the purposes of this introduction could be better served by showing you what was possible first by tools and case studies. If you want to go further, you always can.

This workbook by no means exhausts the topic of digital text analysis, but we hope that you have learned enough to get a sense of the possibilities that such methods can bring. Check out the Further Resources page for other approaches, inspirations, and provocations. If, while reading the book, you found errors or sections that need clarification, please drop a note in our discussion forums or on our GitHub issues page.

Thanks for reading!

Brandon and Sarah

# Further Resources

# Further Resources

Each individual lesson contains suggested further readings on the particular topic discussed in that section. Here we wanted to gather two types of resources. First, we wanted to gather a few more useful tidbits that didn't fit well anywhere but that will be helpful to anyone exploring text analysis. Second, we wanted to point you towards other fantastic tutorials and textbooks for text analysis that go further in depth than we do here. Interested browsers should also check out the lessons on particular topics of interest to make sure you see any and all resources.

## Secondary Readings on the Digital Humanities

- Digital Humanities Zotero Group
- Leary, Patrick. "Googling the Victorians."
- Moretti, Franco. *Graphs, Maps, Trees*.
- Kirsch, Adam. "Technology is Taking Over English Departments: The False Promise of the Digital Humanities."
- LA Review of Books, *The Digital in the Humanities* series
- Rockwell, Geoffrey and Stéfan Sinclair. *Hermeneutica: Computer-Assisted Interpretation in the Humanities*.

## Tutorials and Textbooks

- Arnold, Taylor and Lauren Tilton. Humanities Data in R.
- Bird, Steven, Ewan Klein, and Edward Loper. Natural Language Processing with Python: Analyzing Text with the Natural Language Toolkit.
- Crymble, Adam, Fred Gibbs, Allison Hegel, Caleb McDaniel, Ian Milligan, Evan Taparata, Amanda Visconti, and Jeri Wieringa, eds. The Programming Historian.
- Graham, Shawn, Ian Milligan, and Scott Weingart. *Exploring Big Historical Data: The Historian's Macroscope*.
- Jockers, Matt. Text Analysis with R for Students of Literature.

## Tools

- Google NGram Viewer
- Prism
- Voyant
- Zotero

## Applications of Text Analysis

- Quantifying Kissinger
- Viral Texts
- Syuzhet Part One, Syuzhet Part Two
- Mining the Dispatch
- How a Computer Program Helped Show J.K. Rowling write A Cuckoo's Calling
- Text Analysis of Trump's Tweets Confirms He Writes only the (Angrier) Android Half

- Walmart Gets Its Worst Yelp Reviews in Black and Latino Neighborhoods

## Other

- Networked Infrastructure for Nineteenth-Century Electronic Scholarship (NINES)
- Stanford Literary Lab

# Adapting This Book

# Adapting This Book for Another Course

***The GitBook platform that we use for publishing is changing rapidly. While you can fork our GitHub Repository and edit your own versions of the files, the GitBook platform as of this writing is too unstable for us to develop reliable documentation about how to publish your own version of the text. We will update this page when the issue has been resolved. Until then, the instructions in the Publishing section below should be considered out of date and unstable. If you are able to import your own copy of the text on GitHub by mimicking the instructions below, please make an issue on our GitHub page to let us know.***

We encourage others to use this book for their own courses and to change it to meet the needs of their own contexts. The publishing platform here helps to facilitate this process. We especially imagine people reworking the exercises in each chapter to reflect their own disciplinary content. With a little effort you can rework the book for your own purposes and publish it to GitBooks for your students to use.

**Note:**

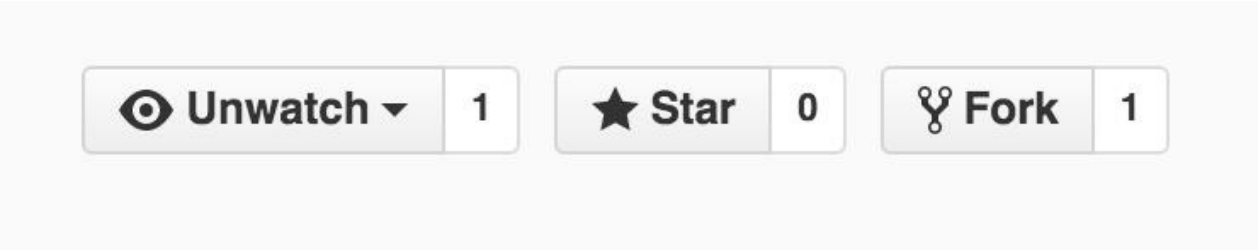
- **Copying the book will only get you a particular version of the book at a particular point in time. By default, any changes we make to the book after you copy it will not be reflected in your version of the book. Syncing your version of the book with ours will likely conflict with any changes you have made, so we would only try that with great care.**

## Getting Your Own Copy

The contents of this book are hosted in a repository on GitHub and rendered to the internet via GitBooks. When we make changes to the file structure hosted on GitHub, the changes populate out to our GitBooks account, which renders the various files into the web version of the book. To make your own remixable copy of the book, you will need to make a copy our GitHub repository and sync your copy with a GitBook of your own. Things you'll need to begin:

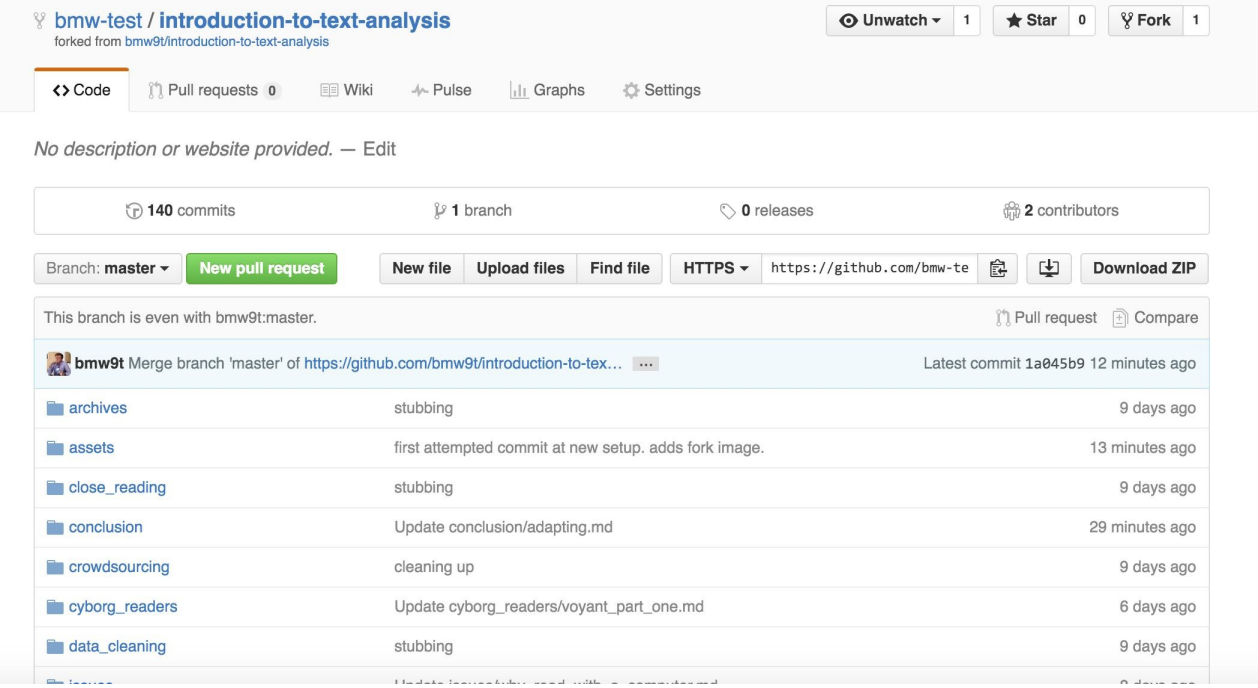
- GitBooks Account
- GitHub Account
- GitBooks Editor (optional depending on your command line and markdown fluency)

First you will need to make a copy of our GitHub repository for your own account. When logged in and looking at our repository page, you should see these three buttons in the top-left corner of the window:




**Forking** is Github's term for creating a copy of a repository for yourself - imagine a road forking and

diverging into two paths. If you click fork, GitHub should start the copying process. When finished, you will be redirected to your fresh copy of the repository.

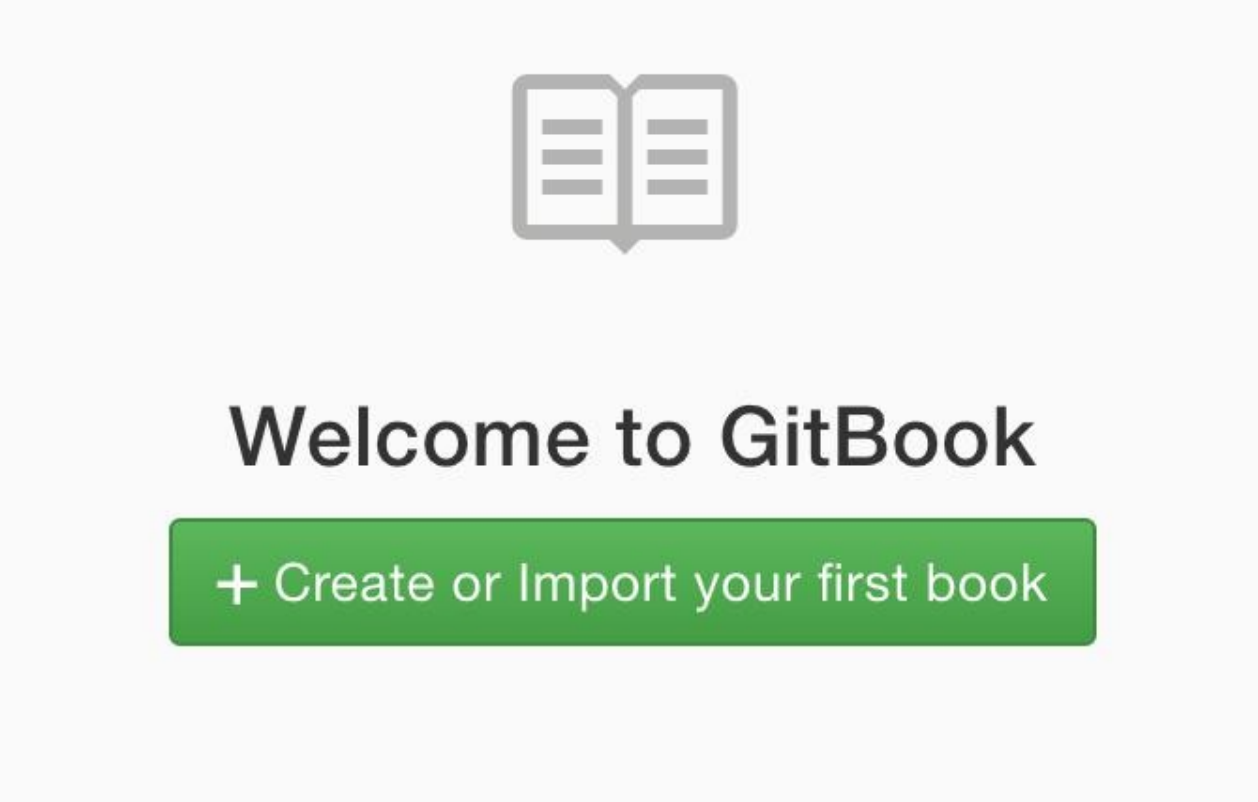



Note the "forked from bmw9t/introduction-to-text-analysis" statement at the top of the window, which lets you know where the book originated from. Above that you will see your own book's location. You now have your own version of the book's various files, and any changes you make to your own version will not affect our original book. GitHub will also keep track of your book's history for you.

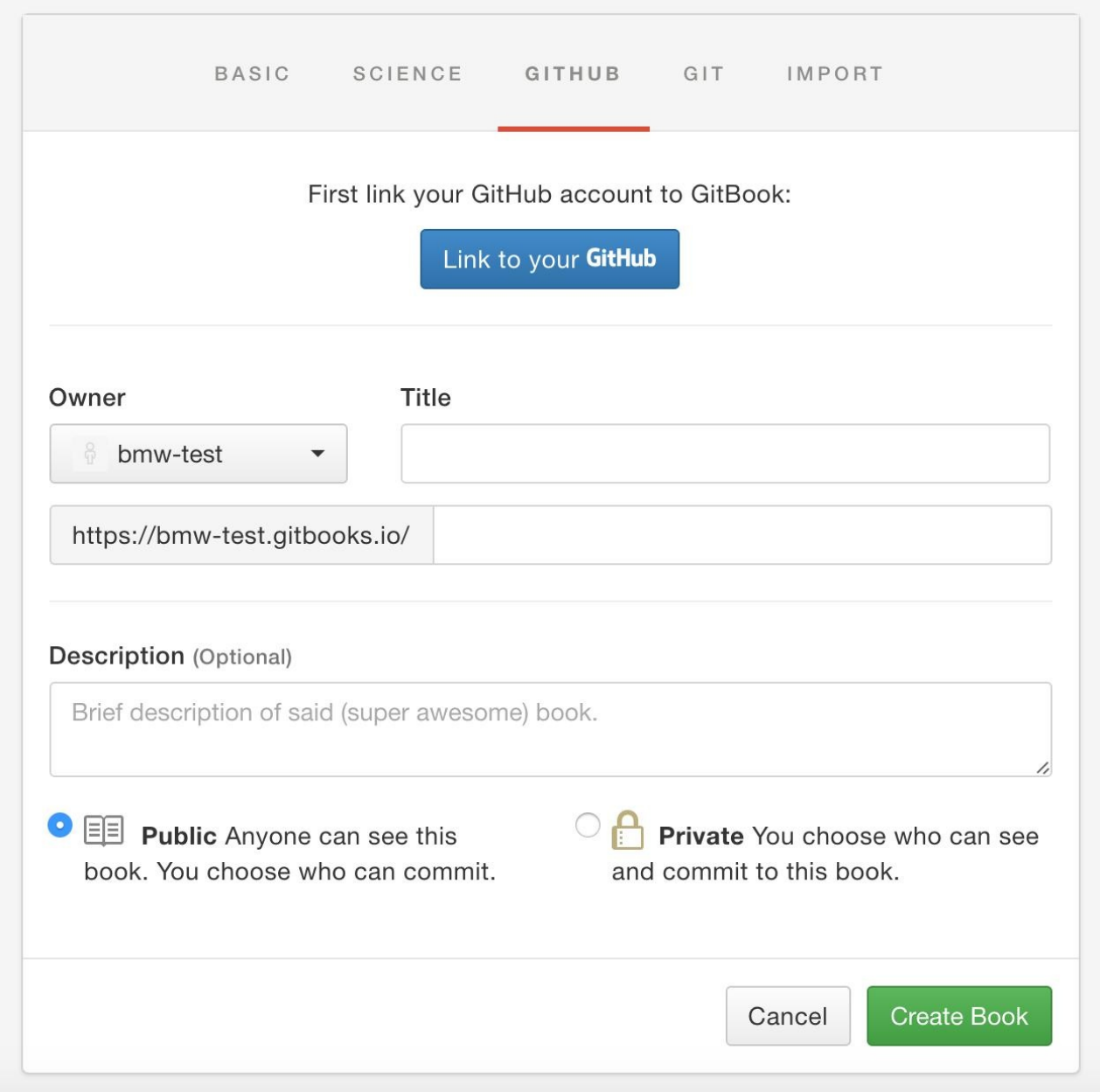
# Publishing

**Note: GitBook is still under heavy development, and these steps might have changed since last writing.**

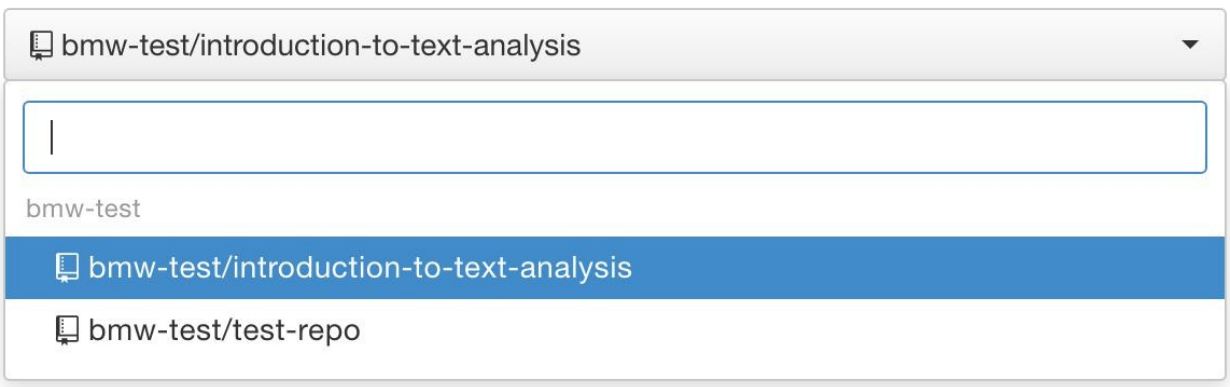
You have a copy of all the files that make up the book, but you will need to sync them with GitBooks if you want to publish them online in the same way that we have done here. To do so, after logging into GitBooks you will click on the green 'Import Button.'



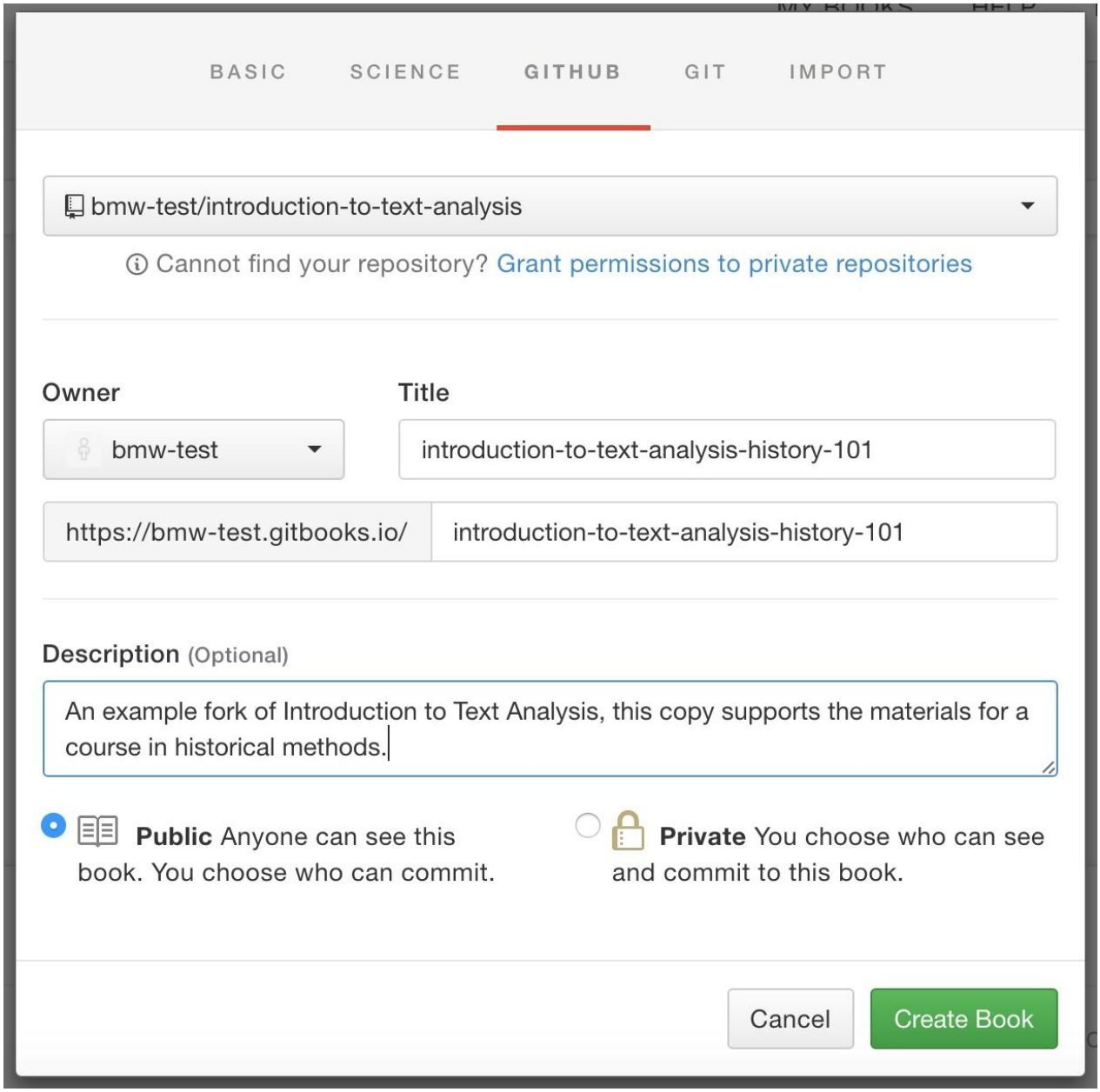

Selecting the "GITHUB" option, you will need to link your GitHub account and verify your account by an email.




After linking your GitHub account, if you have more than one respository under your name you will have to select the one that you want to import to GitBooks. In this case, we will import the *Introduction to Text Analysis* repository.




Give your repository a name and a description, and you're all set. A complete form should look something like this:




You now have a working copy of the book hosted on GitHub and rendered in GitBooks (GitBooks should automatically redirect you to your copy). You can do anything you want with these files, and they won't affect our own base copy of the resources.

# Editing

## Markdown

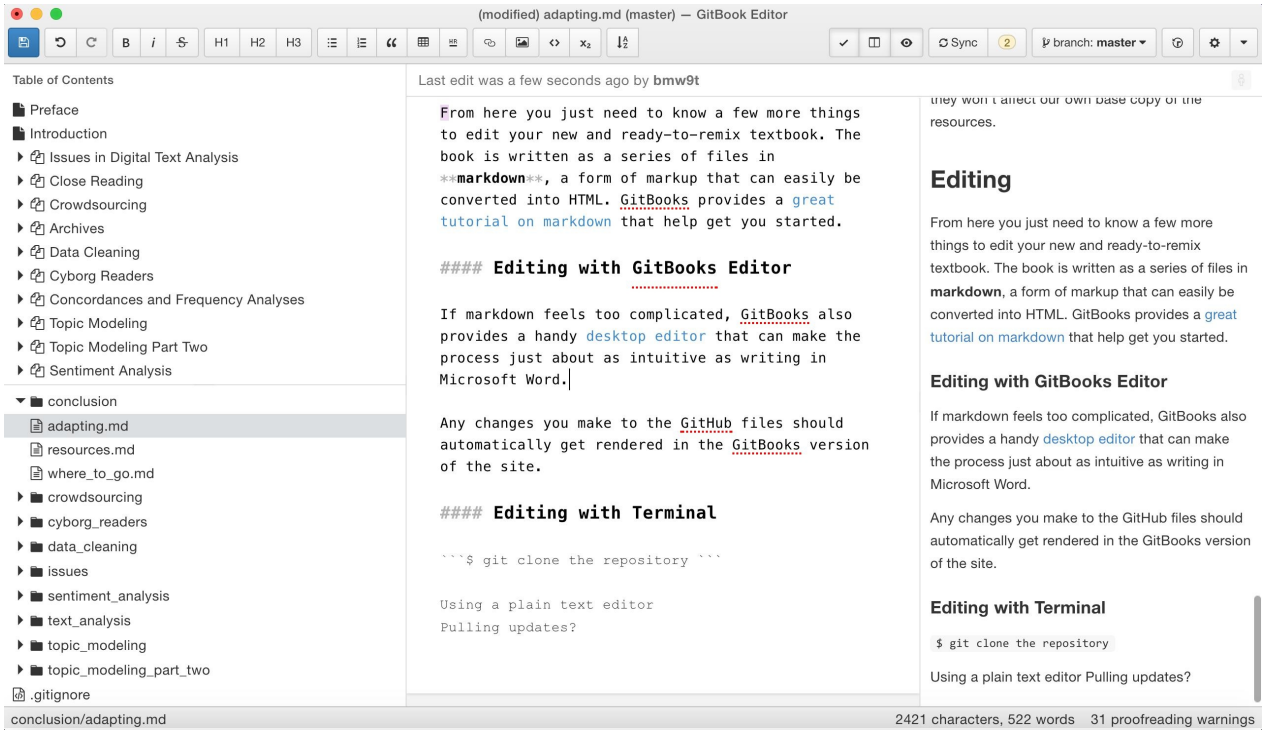
From here you just need to know a few more things to edit your new and ready-to-remix textbook. The book is written as a series of files in **markdown**, a form of markup that can easily be converted into HTML. GitBooks provides a great tutorial on markdown that help get you started.

## Editing with GitBooks Editor

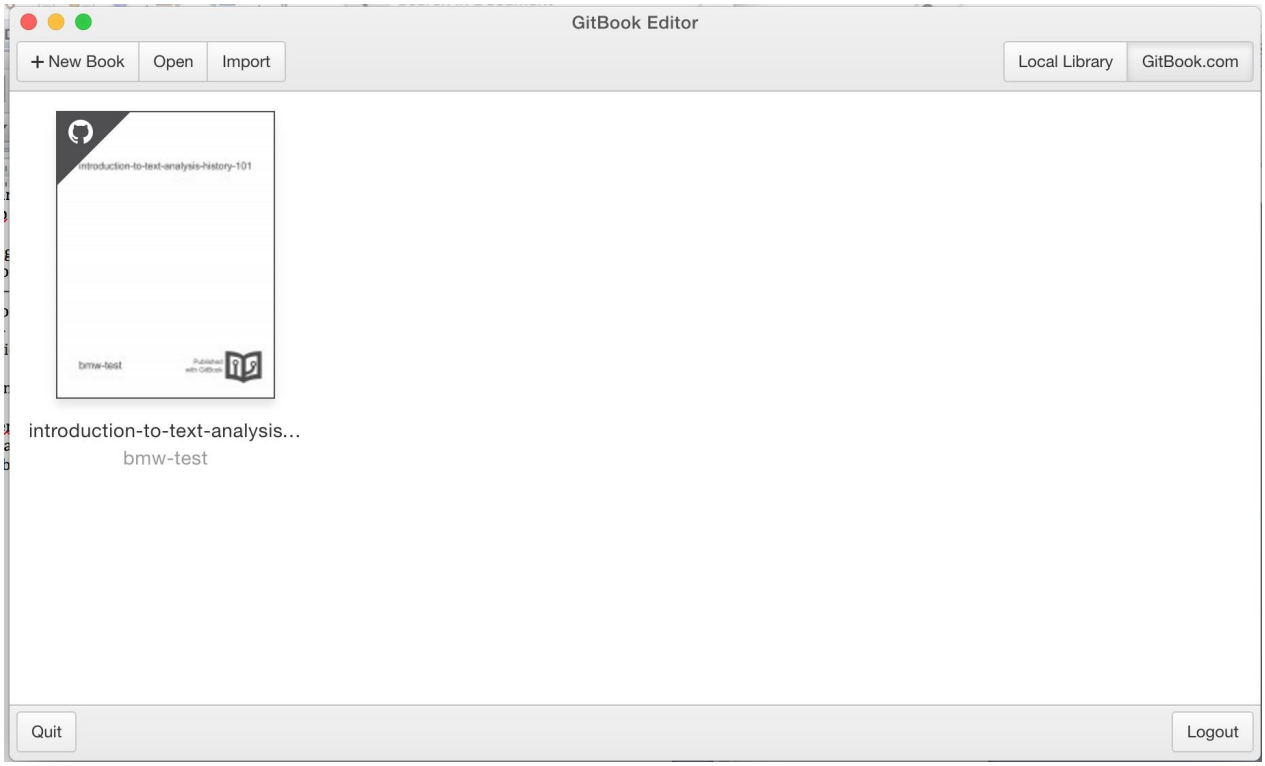
If markdown feels too complicated, GitBooks also provides a handy desktop editor that can make the process just about as intuitive as writing in Microsoft Word. You can type in markdown, but the editor will also convert certain commands to markdown for you:

**bolded text** will render as **bolded text**.

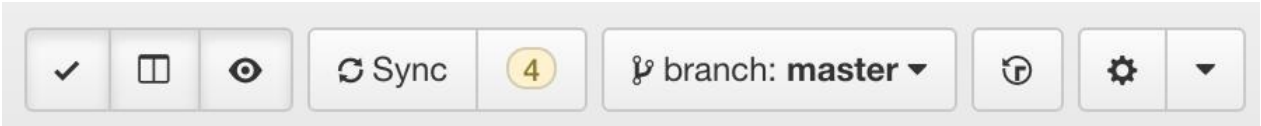
But you can also highlight text and press command + b as you would in Microsoft Word to produce the same effect.




The interface provides a preview of what your text will look like to the right of the window, which can be very helpful if you are new to markdown. If you decide to work in the GitBooks Editor, you will need to log in the first time you do so. Then select the "GitBooks.com" option for importing.




The computer will **clone**, or copy, the book to your computer. From there, you can follow the instructions in the editor's documentation. The only significant difference from MS Word is that, after saving your work, you will need to click the sync button to upload your content to GitHub.




After doing so, any changes you have made from the GitBooks editor will also change the GitHub repository's files, which will then automatically get rendered in the GitBooks version of the site. You are all set!

**Editing with Terminal**

If you are planning to use terminal, the process is fairly similar. Once you have forked and have your own copy of the book on GitHub, you will just clone it to your computer using the clone url found at the top of your repository's page on GitHub. Here is the one for the original book:


Find your own clone url, copy it to your clipboard, and use it like so (without curly braces):

```
$ git clone {your_clone_url here}
```

This will copy the repository to your machine. From there, you can edit using a plain text editor as normal and make changes to the repository using git.

At this point you should have everything you need to edit your copy of the book as you see fit for your own needs. If we haven't covered something here or you run into problems, drop us a line in our

discussions forum.