Teaching Humanistic Data Analysis

Ryan Cordell
Associate Professor of English
Northeastern University, USA
r.cordell@northeastern.edu

Abstract: *This paper advocates for a "data humanism" that is distinct from both computer science and data science. Seeking to simply teach data science skills to humanities students misses the unique contributions our fields and our students might make to broader conversations about data and culture. The most important reason to imagine a distinct data humanism is that doing so in turn imagines students who will go into the world outside the academy and change it, rather than simply slotting into existing frameworks. This data humanism should be exploratory, iterative, and dialogic, and takes as its goal directing scholars' attention to unexpected places in and outside digitized collections, while raising new questions about those collections and their absences. The paper concludes by offering four concrete recommendations for fostering humanistic engagement with data in the undergraduate and graduate classroom. These comprise starting with creativity, teaching using humanistic rather than pre-packaged data, foregrounding the investigation of corpora over computational methods, and likewise foregrounding the inculcation of a mindset for approaching data over any particular method.*

Keywords: data ■ computational analysis ■ programming ■ pedagogy ■ data science ■ digital humanities

INTRODUCTION: HUMANITIES DATA

Lorraine Daston and Peter Galison's 2017 book *Objectivity* attempts to trace the emergence of objectivity as a concept, ideal, and moral framework for scientists during the nineteenth century. The work focuses primarily on shifting ideas about scientific images during the period. In the eighteenth and early nineteenth centuries, Daston and Galison argue, the scientific ideal was "truth-to-nature," in which particular specimens are primarily useful for the ways in which they reflect and help construct an ideal type: not this leaf, specifically, but this type of leaf. Under this regime scientific illustrations did not attempt to reconstruct individual, imperfect specimens, but instead to generalize from specimens and portray an ideal type.

The "truth-to-nature" framework changed, in part, because of the introduction of photography, which nudged scientists toward a new ideal of mechanical objectivity. In early debates about the virtues of illustration versus photography, illustration was touted as superior to the relative primitivism of photography, as technologies such as drawing or engraving simply allowed finer detail than blurry nineteenth-century photography could. Nevertheless, photography increasingly dominated scientific images over the course of the century because it was seen as less susceptible to manipulation, and less dependent on the imagination of the artist (or, indeed, of the scientist). As Daston and Galison explain,

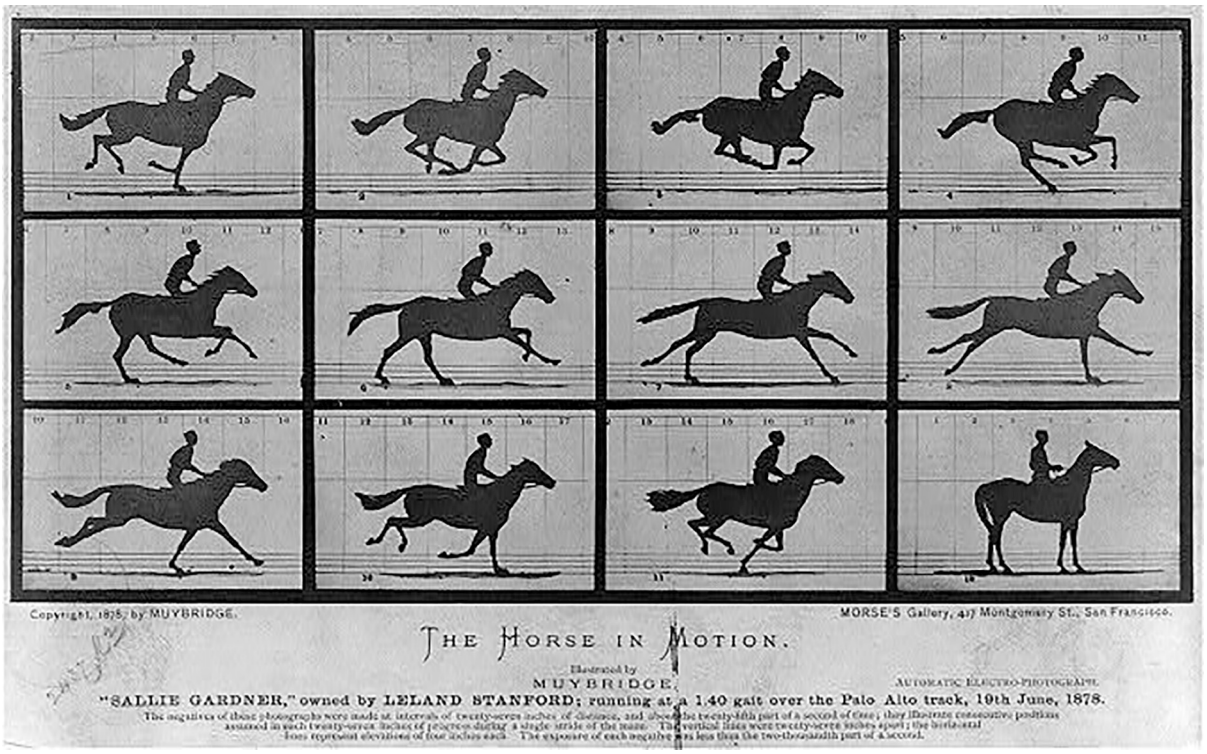
> It is this internal struggle to control the will that imparted to mechanical objectivity its high moral tone.… One type of mechanical image, the photograph, became the emblem for all aspects of noninterventionist objectivity.… This was not because the photograph was more obviously faithful to nature than handmade images—many paintings bore a closer resemblance to their subject matter than early photographs, if only because they used color—but because the camera apparently eliminated human agency. (187)

For scientists increasingly worried that their own stubborn wills would sully the truth of their findings, mechanical means of image production offered a solution.

Eventually, the camera also enabled views of nature impossible for the human eye alone. As the exposure time required for photography was reduced, for instance, it enabled new accounts of motion. Consider Eadweard Muybridge's famous photographs of racing horses, made between 1878 and 1884, which established that horses do indeed bring all four hooves off the ground at certain moments during a gallop. In this and subsequent photo series, the technology of the photograph dramatically recontextualized scientists' conception of nature by allowing them to freeze motion and analyze its constituent "parts" separately. In other words, these photographs convert a fluid motion, or a process, into data: discrete units of observation, measurement, or analysis.

This moment in scientific history recalls a touchstone essay for those engaged in computational text analysis in the humanities. In "Text: A Massively Addressable Object," Michael Witmore argues that "What distinguishes th[e digital] text object from others" is that

> it is *massively addressable at different levels of scale*. Addressable here means that one can query a position within the text at a certain level of abstraction.… The book or physical instance, then, is *one of many levels of address*. Backing out into a larger population, we might take a genre of works to be the relevant level of address. Or we could talk about individual lines of print, all the nouns in every line, every third character in every third line. All this variation implies massive flexibility in levels of address. And more provocatively,



Eadweard Muybridge's "The Horse in Motion," late 1878. COURTESY OF THE LIBRARY OF CONGRESS

> when we create a digitized population of texts, our modes of address become more and more abstract: all concrete nouns in all the items in the collection, for example, or every item identified as a "History" by Heminges and Condell in the First Folio. Every level is a provisional unity: stable for the purposes of address but also stable because it is the object of address. Books are such provisional unities. So are all the proper names in the phone book.

As Witmore acknowledges, "Physical texts were *already* massively addressable before they were ever digitized" through scholarly apparatus such as indices, concordances, or marginal notes. It is the flexibility of address that is new. Suddenly the book, or even the library, needn't be converted into a new form to become data: the book or library can itself be data.

Certainly other forms of humanistic data also existed before computation: consider the library card catalog or subject bibliography as forms of database. Like Muybridge's photographs, however, the computer reshapes our relationship to text by making it radically separable and manipulatable: a unity that can be reshaped into its constituent parts and at many levels, as Whitmore outlines. The primary objects of much humanistic inquiry—and text is only one of these, though it is what I know best and thus what I focus on here—can be considered as discrete units of observation, measurement, or analysis.

TOWARD A DATA HUMANISM

This paper poses and begins to answer two interrelated questions: first, what precisely do we mean by "data analysis" as an area of instruction in humanities courses? and second, how might we distinguish humanistic data analysis from analogous methods in the natural and social sciences? In much the same way that Datson and Galison argue scientists turned toward machines in order to avoid the mistakes of human judgment, in some articulations of digital humanities, computation is invoked as a solution to problems of will that are quite familiar from decades of humanistic scholarship. We all know that the canon of English literature was shaped as much by the identities of its gatekeepers as by the inherent virtues of the books it includes. Feminist, postcolonial, and other theoretical schools have taught us that our biases limit our critical horizons and result in atrophied canons that do not adequately represent human identities or literatures.

Methods such as distant reading, macroanalysis, or cultural analytics are sometimes proposed as methods that can bypass the fallible human wills that constructed existing canons through a species of mechanical objectivity. While human beings choose what to focus on for all kinds of reasons, including conscious or unconscious biases we would seek to mitigate, the computer will look for textual patterns unencumbered by those preconceptions. Under this vision of computational research, the machine is less susceptible to the social, political, or identity manipulations of canon formation.

If we look closer, of course, we see that this position only sets the problem at one remove, in the humans writing the code rather than in the humans assembling the canon. As scholars such as Safiya Umoja Noble,[1] Cathy O'Neil,[2] and Virginia Eubanks[3] have amply shown, computer programs always come laden with their creators' biases and oversights. Whether purposefully or inadvertently, software designed to create business or government efficiencies often reinscribes structural inequalities such as white supremacy, and as scholars such as Ben Fagan (2016) have demonstrated, the same holds true for many mass digitization projects that through their selection processes reinscribe the dominance of mainstream cultures. Given this reality, I would suggest that we cannot hope to automate our way into a more just canon formation. Instead, we should advocate for a humanities data analysis (HDA) that seeks primarily to point our attention to the gaps in our digitization and critical practices. To my mind, the most compelling articulation of this potential remains Lauren Klein's description of topic modeling (2015) as "a technique that stirs the archive" and which, in an iterative relationship, can be in turn stirred by the archive (and as a side note, Klein's book-in-progress with Katherine D'Ignazio, *Data Feminism,*[4] will be an essential touchstone in this area when complete).

In Klein's 2015 account, methods of humanistic data analysis allow researchers and students to articulate connections or paths through a collection that are not apparent through linear reading, while at the same time—and this is essential—the contours of our collections shape the kinds of connections or paths available, such that "domain experts … must be able to probe the semantic associations that the model proposes." In order to make sense of a topic model, the researcher must understand the collection upon which it was trained in order to articulate the set of possibilities available to the model and to notice what possibilities are missing.

The primary outcomes of the computational analyses Klein describes are not statistical models—though they might employ such models along the way—but instead a speculative dialogue between researcher and data. We can trace a similar conception of humanistic data analysis in Stephen Ramsay's book *Reading Machines* (2011a). Ramsay is best known for a provocative three-minute position paper, "Who's In and Who's Out" (2011b) delivered at the 2011 MLA conference—in a roundtable, it must be remembered, in which all participants were asked to draft provocative position papers to spur discussion—in which he argued that digital humanities scholars should know how to code. In *Reading Machines*, however, Ramsay offers a deeply nuanced—and, frankly, more characteristic—vision of what might constitute an "algorithmic criticism" in literary studies. In particular, Ramsay rejects scientistic frameworks for computational work in favor of a dialogic framework:

> If text analysis is to participate in literary critical endeavor in some manner beyond fact-checking, it must endeavor to assist the critic in the unfolding of interpretive possibilities.… (10) However far ranging a scientific debate might be, however varied the interpretations being offered, the assumption remains that there is a singular answer (or a singular set of answers) to the question at hand. Literary criticism has no such assumption. In the humanities the fecundity of any particular discussion is often judged precisely by the degree to which it offers ramified solutions to the problem at hand. We are not trying to solve Woolf. We are trying to ensure that discussion of *The Waves* continues. (15)

In these passages Ramsay emphasizes the distinct hermeneutic underpinnings of humanistic work. In this paper I will follow Klein and Ramsay's frameworks to argue for humanistic data analysis that is primarily exploratory, iterative, and dialogic, with a goal of directing our attention to unexpected places in and outside our digitized collections, while raising new questions about those collections and their absences.

This "data humanism" is not entirely dissimilar to the methods often gathered under the phrase "data science." At least in the United States, new masters programs in data science seem to pop up daily, alongside promises about the rewarding and well-compensated work that will follow such a credential. From the rhetoric alone, data science seems to be something like, but not quite identical to, computer science: programming is required, likely in a statistics-friendly language such as R, but the focus is not on building systems or writing software. Instead, the data scientist is adept at "just in time" programming that explores economic, political, or civic datasets, identifying trends of immediate use to government, corporate, nonprofit, journalistic, or other actors. The key trait of the data scientist seems to be flexibility, as they work across domains and datasets seeking meaningful patterns. Though these qualities are also useful for humanities data analysis, I will propose that data *humanism* must differ, both in substance and methodology, from data analytics as it is imagined and practiced.

Digital humanities scholars have been routinely accused of trying to turn literature, history, and related fields into branches of computer science. This charge is largely untrue. In particular, the interpretive aims of digital humanities (DH) work are quite distinct from computer science (CS), and many claims that conflate the fields stem from mistaken assumptions about what computer scientists do. However, the ways we teach computing proficiency, the specific computational techniques we employ, and the ways we describe the findings of computational work have drawn too exclusively from corpus linguistics, data science, and related social sciences. We have largely failed to develop specifically humanistic approaches to computational data analysis, and the keen lack of such methods has become a particular hindrance to our field's development. Seeking to simply add data science skills to humanities students misses the

unique contributions our fields and our students might make to broader conversations about data and culture. The most important reason to imagine a distinct data humanism is that doing so in turn imagines students who will go into the world outside the academy and change it, rather than simply slotting into existing frameworks. I do not propose data humanism as opposed to data science, but as a complement and (occasional) corrective.

IN PRACTICE

So what might data humanism look like in practice, and how do we inculcate its skills in our students? The remainder of this paper will draw on my experiences teaching data analysis to both undergraduate and graduate students in Northeastern University's English Department, largely through courses such as Reading and Writing in the Digital Age,[5] Technologies of Text,[6] Humanities Data Analysis,[7] and Reading Machines: Technology and the Book.[8] These issues are also very much on my mind as I prepare to teach a version of Humanities Data Analysis,[9] at the 2019 Digital Humanities Summer Institute at the University of Victoria (Canada). The principles I draw from those classroom experiences evidence my own evolving sense of how a data-rich humanities pedagogy might be structured and what goals it might seek to meet, which is to say I will reflect as much on approaches that have failed as on those that have succeeded. The *ideals* I outline are not ones I believe my teaching has yet met, though I seek each semester to work closer to them. In outlining these ideals, I also do not mean to imply that no one in computer science, data science, digital humanities, or cultural analytics follows these practices. Certainly many do; my recommendations are nearly all grounded in examples (which I will try to cite fairly).

1. START WITH CREATIVITY

For many years, I would begin the programming units in my classes with text analysis: word counts, keywords in context, n-grams, topic modeling. After spending several weeks (in a single unit within a larger class) or most of the semester (in an HDA focused class) on these activities, we might devote a final week to a more creative engagement with programming, such as building a poetic Twitter bot. In recent years, however, I have become convinced that this is precisely the wrong way to introduce humanities students to programming or data. This past year, I reversed the order of operations, and I think all of my classes benefited immensely.

In my Reading and Writing in the Digital Age course, we began our unit on data with Giorgia Lupi and Stefanie Posavec's project *Dear Data,*[10] which began as a website and became a beautiful book. Lupi and Posavec spent a year recording a different aspect of their daily lives each week, creating a new way to visualize that data on a postcard, and mailing the postcards to each other; the book compiles these 104 experiments and presents them beautifully. We began here because I wanted my students first to understand that data is a bigger category than digital data, and that the suites of visualization tools built into existing computer programs include only a handful from the infinite possibilities for data visualization. Students were then charged[11] to spend a week recording and then visualizing some aspect of their own data, reflecting both on what they learned from the data, the choices they made in representing it, and the ways those representational choices shaped what they could (and couldn't) learn about the data. An exercise such as this prepares students to encounter computational data work from a productively critical perspective.

This year I also began the programming units in two courses, my undergraduate Technologies of Text and my graduate Reading Machines, with the Twitter bot assignment that used to serve as the cap to this unit. On the whole the exercise is pretty simple: students choose a poem or other short text and then build a program that will substitute words of the appropriate part of speech into the chosen poem: think of the children's game Mad Libs. The results are sometimes nonsensical, sometimes hilarious, and sometimes oddly profound. Practically, students learn a number of important skills through this lesson, including how to manipulate text strings in R, how to query web services using an application programming interface (API), and how to output the results of an R program back to a web service such as Twitter. Most importantly, however, they

learn these skills through a process that is generative and creative, and which leverages their existing skills, such as the ability to understand and analyze a poem: at least well enough to know what substitutions are likely to "work" and be funny. Though simple, this exercise fits into a long tradition of cut-up and experimental poetry with which students are familiar, and it helps them understand programming from the outset as something more creative and expressive than they might have imagined.

The way we introduce data analysis matters. If you survey publicly available "learn to code" courses on sites such as Coursera or Codeacademy, you will notice that, regardless of language, they all begin in nearly the same way: teaching students how to do math. I do not want to imply that math does not enter into humanities data analysis—though I will discuss this further below—but I will argue that, given the huge range of tasks to which programming can be put in 2019, beginning in this way is not necessary and can throw up immediate barriers to humanities students who perhaps see themselves as not skilled at or interested in mathematics.

Most current programs for teaching coding or data analysis presume that the endeavor must be rooted in computer science, but I follow Annette Vee in seeking to decouple programming as a practice from computer science as a discipline, to the benefit of both. In her 2017 book *Coding Literacy*, Vee argues that in much of the world today "computer code is infrastructural," a literacy in the ways it is valued and deployed: "layered over and under the technology of writing, computer code now structures much of our contemporary communications" (3).

> When we consider programming a mode of written communication, it is no longer bounded by the field of computer science. Its roots are no longer solely in math, engineering, and science; they include written communication as well. Decoupling programming from CS not only helps us understand programming as communication but also frees CS from being overly identified with just one of their practices … neither programming nor CS is well served by the idea that they map perfectly onto each other. (41)

Vee shows in her book that "programming never was a domain exclusively for specialists" while advocating that we consider programming a literacy that can be applied in many ways and in many domains. This is also my goal, and is a primary reason that I seek to show students that a humanistic approach to data can have distinct aims from the outset. To expand out a bit: we need new pedagogical resources to teach data analysis for humanities students that begins with questions, data, and programming tasks that will resonate with those students.

2. TEACH USING DOMAIN-SPECIFIC DATA

My next recommendation is straightforward: when teaching students humanities data analysis, we should use humanities data. This point might seem obvious, but it has not been so in practice, in part because it can be harder to practice than it should be. Insisting on teaching with humanistic data leads inevitably to a complementary idea, that we should not shy away from data that is complex and messy. Computer science and data science are often taught using tidy datasets that are designed, quite literally, to help students gain proficiency with particular methods. I think of the "mtcars" data that is built into the R programming environment. It is a relatively small, well-structured table of information about various car models, including details such as their miles per gallon efficiency. The dataset was extracted from the 1974 issue of *MotorTrend* magazine, and is designed to be a resource for learning how to use R's mathematical or plotting functions, and thus is invoked in tutorial after tutorial teaching just these functions. The other most popular built-in datasets for R include "iris," which includes various measurements for 50 flowers in the iris family; "ToothGrowth," which contains the results of an experiment studying the effect of vitamin C on tooth growth; or "USArrests," which includes statistics about violent crime in the US.

There are a number of problems with defaulting to such datasets when teaching humanities students. First, our students often find it difficult to extrapolate from data about cars or tooth growth to the domains they care about. This is especially true for students coming to data analysis with some hesitancy or trepidation, as many stock training datasets available resonate, teleologically, as corporate or governmental. To state an even stronger version of this point, these

datasets look and feel like what our students think data should look and feel like: which is to say, they look and feel like information from domains far removed from humanistic inquiry, and they alienate students rather than welcoming them in.

Second, however, such datasets are ontologically distinct from humanities data, insofar as data drawn from literary, historical, and related domains is defined (at least in part) by its complexity and messiness. This is in fact why many computer science professors are drawn to our data for their own research: it is intriguingly difficult. A few years ago, I co-taught a course titled Bostonography for computer science undergraduates at Northeastern. Our students in that course repeatedly expressed how frustrated but exhilarated they were to be working with what they called "real data"—which is to say, historical data related to the city of Boston—rather than the canned datasets they encountered in their CS classes. The data in our class, we heard, was uniquely challenging but more rewarding than the dull stuff they encountered again and again. Humanities students need to work with this challenging humanities data from the beginning.

Unfortunately it is not self-evident how to teach using humanities data—so much so that perhaps the most common complaint among those teaching such classes is the lack of relatively small, teachable data sets in literary studies, history, or related fields. Many of the datasets one encounters in the field's literature are simply too large for a group of students to work with simultaneously in a classroom setting, and in fact can be frustrating even outside the classroom when it takes many minutes—or much longer—each time a student has to learn that a line of code does not work as it should. The iteration required for learning programming—run the code, see if it works, revise, run it again—cannot happen with very big data, which is why fields such as CS and data science have developed the teaching datasets I described earlier. As more people seek to cultivate data humanism in students, we will need to cultivate also manageable, but still interestingly messy, datasets for use in classrooms.

My own research largely centers on historical newspapers, and I have developed some unexpectedly generative teaching exercises around the Library of Congress's *U.S. Newspaper Directory.*[12] This is not their collection of digitized newspapers, which is far too large for classroom work, but instead their index of metadata about all known newspapers founded in the United States between 1690 and the present. Like R's "cars" dataset, there are many natural experiments to be run here to learn how to compare categories or plot: how many weeklies versus dailies were founded in the 1870s? Which states had the most newspapers at the turn of the century? Can we analyze or visualize the most popular words used in the titles of new newspapers over decades, and what might such analyses tell us about the changing ways editors and readers conceptualized the newspaper across time? Not all of my students are newspaper researchers, but because this data is about something closer to their interests, they know the kinds of questions we might ask about it, and these investigations are more easily transferred into their own domains.

3. FOREGROUND CORPUS OVER METHOD

Asking questions about the data we use in our classrooms leads directly to my next recommendation, which is that in humanities data analysis, teaching our corpora or datasets should precede teaching methods. In her 2018 book *A World of Fiction*, Katerine Bode draws on her book history expertise to critique the ways many computational literary scholars approach the data at the heart of their analyses. In particular, Bode argues that

> in their literary-historical work both [Franco] Moretti and [Matthew L.] Jockers present literary data and digital collections as pre-critical, stable, and self-evident. In conceiving data and computation as providing direct and comprehensive access to the literary-historical record, they deny the critical and interpretive activities that construct that data and digital record and make them available for analysis. (20)

Bode does not claim distant reading scholars fail to disclose the corpora from which their conclusions are drawn, but instead that they often do not adequately describe those corpora: what they include, what they do not. In this way, Bode argues, distant reading looks a lot like new

critical close reading, in that the specific material circumstances of the text at hand—whether that text is a single Keats poem or the HathiTrust library—are discounted or overlooked in favor of an idealized notion of "the text" or "the archive."

Bode's claim extends critiques by scholars such as Johanna Drucker, who argues that the very word "data" implies that the information it labels is "a 'given' able to be recorded and observed" or simply "a natural representation of pre-existing fact" (Drucker 2011). While I am not *quite* advocating with Drucker that we replace the word "data" with "capta"—because "*capta* is 'taken' actively"—I would concur that a data humanism must begin with discussions of data construction rather than data analysis. To make claims from the Library of Congress's *Chronicling America* newspaper archive, as we do in the Viral Texts project,[13] requires understanding how its newspapers were compiled through the National Digital Newspaper Program, which lays out specific selection guidelines for states that participate. These guidelines include ideas of "representativeness" around circulation and influence that ultimately shape the kinds of newspapers included, and the kinds excluded, from the overall collection.

Returning to classroom conversations about the Library of Congress's *Newspaper Directory*, I would note that conversations about what we might learn analyzing it lead naturally to rich conversations about the construction of the dataset itself. Who compiled these lists? How did they obtain the information about the newspapers? What information is not included? For instance, does the data include anything about the communities that various newspapers served? If we analyze the most popular words used in titles, what contours of the newspaper landscape in those years are we necessarily excluding? A humanistic approach to data analysis must foreground such conversations and teach students how to probe the intellectual, social, and political underpinnings of each dataset or corpus they encounter.

4. FOREGROUND MINDSET OVER METHOD

The courses I draw from in this paper largely teach humanities data analysis through coding in the programming language R. I would not argue that coding is the only way that effective humanities data analysis might be taught or practiced, in large part because this raises the barrier of entry too high. This is an area of inquiry that needs more voices, not fewer. There are many good tools out there for performing most common analysis tasks, and one can be thoughtful about all of the issues I have thus far outlined in graphical user interfaces (GUIs) as easily as in a programming language. And to be frank: my own work in this area began using GUIs and applications, which helped me cultivate approaches to questions that eventually required me to cultivate programming abilities. That early work was no less valid than the work I do now.

However, I will outline the reasons that I have moved more and more toward approaching humanities data analysis in my classes through programming rather than using common DH applications. In fact I should immediately modify that to say that I teach humanities data analysis through workbooks, which are in a way an intermediary form between an application and pure programming. In R such workshops are composed as RMD files,[14] and they weave together prose with executable blocks of code. In this form I can include relatively complex chunks of code, even in the first days that we work on programming, so we can move immediately to applications that will feel more consequential to students than printing "Hello, World" might. When looking at an RMD file like this, students might not understand each line. In contrast to a GUI, however, in this form each line is available for inspection, and good deal of the work we do together, at least initially, consists of running a block of code, looking at the results, and then working back through the code to understand what precisely it did step-by-step.

I follow this procedure for a few reasons. First, I largely concur with my colleague, historian Ben Schmidt (2016), that "digital humanists do not need to understand algorithms *at all*," insofar as understanding means grasping precisely what each Greek letter in an algorithm's equations stands for or being able to precisely reproduce the underlying math. What humanists do need, Schmidt continues, is "to understand the transformations that algorithms attempt to bring about." If students are to use computational techniques responsibly, they need to understand what happens to text or tabular data to move from a set of, say, novels to a vector space model. What as-

sumptions are baked into the method? What variables can be controlled, and what happens when they are changed? It is possible to do this work within a GUI? Gale's own Digital Scholar Lab is quite robust and good at foregrounding explanations of the methods it includes and the variables that researchers can control within it.

I do not wish to pretend that working in a programming language like R lays everything bare, given that many elements of particular algorithms are packaged within functions that can be run with little understanding. For me, however, working through the code helps students understand a little more fully the processes through which transformation happens, to see each major step in a given process, and to begin asking questions about how particular functions operate. My workbooks typically include chunks of prewritten code as well as instructions about lines that students can manipulate, as well as empty code blocks they can use to copy, paste, and modify the code I have given them. This kind of tinkering introduces a sense of code as a medium—it is literally text that can be copied, pasted, edited—and opens up to later workbooks in which students have more direct agency, as well as independent projects in which students ask questions about their own data.

Here's an important point to make about all of the pedagogy I have been describing: I cannot turn students into proficient, independent programmers in a four-week unit, or even in a full semester class, and that is not my goal. Nor is my goal to bring them to expertise in any particular method: e.g., classification or topic modeling. What I am trying to cultivate is a mindset for approaching data, exploring it, figuring out what questions computation might help answer about it, and then determining what methods might help answer those questions. Ultimately I want them to begin to understand how to outline a series of steps—an algorithm, in a way, though not expressed mathematically—that would allow them to transform the data from its original form into a form pertinent to the question at hand. To actually make that process happen would likely require more research and potentially collaboration. But I would argue the hardest skill for a humanities student to acquire is programmatic thinking, not programming.

A deeply humanistic, exploratory approach to data will not only result in students able to use methods developed for computer science or social science on humanities data, but also will result in the need for new methods required by our materials. This, I would argue, is an area largely unexplored in digital humanities or adjacent fields. We have seen quite sophisticated projects that apply methods such as vector space analysis or topic modeling to novels or historical newspapers, but we have not seen new methods emerging from and specifically for historical, literary, or related data. But this is the future I want to teach toward. One reason we need humanistic data analysis is that we cannot cede either the maintenance and algorithmic framing of digital cultural heritage entirely to engineers or, with all deference to our hosts today, to corporations. Certainly products such as the Gale Digital Scholar Lab (DSL)[15] can be useful starting points for research within their collections and to teach the affordances of the methods they incorporate. Gale's DSL is particularly powerful and flexible; I have been genuinely impressed with its capabilities.

Nevertheless it remains true that no standard suite of analytical tools, however thoughtfully developed, will capture the extent of what we might hope to learn from a historical, literary, or social dataset, and one of the great challenges of humanistic work is that the questions of interest are difficult to generalize or predict well enough in advance to develop a generic tool that will cover them. Teaching students using only out-of-the-box tools will limit their possibilities. In addition, I teach knowing that the majority of my students are not heading toward jobs at well-resourced universities that are likely to subscribe to the databases that make a product like the DSL useful. The majority of them are heading toward work in the much broader field of US universities and colleges or into work in the private sector. If these students are to continue data-rich humanistic work—and, essentially, train their own students to do the same—they will do so primarily using open access tools—including both GUIs and programming languages—and open access data. My students need to understand the basic underpinnings of such tools and be able to flexibly adapt to different institutional situations and, if necessary, know how to begin assembling their own suite of tools for the research and teaching they need to do.

NOTES

1. https://safiyaunoble.com/.
2. https://mathbabe.org/.
3. https://virginia-eubanks.com/.
4. https://bookbook.pubpub.org/data-feminism.
5. https://f18rwda.ryancordell.org/.
6. https://s19tot.ryancordell.org/.
7. http://s17hda.ryancordell.org/.
8. https://s19rm.ryancordell.org/.
9. http://www.dhsi.org/courses.php#DataAnalysis.
10. http://www.dear-data.com/theproject.
11. https://f18rwda.ryancordell.org/assignments/dear-my-data.html.
12. https://chroniclingamerica.loc.gov/search/titles/.
13. https://viraltexts.org/.
14. Examples of RMD files from my recent DHSI class are here: https://github.com/rccordell/DHSI-HDA/tree/master/exercises.
15. https://www.gale.com/intl/primary-sources/digital-scholar-lab.

REFERENCES

Bode, Katerine. 2018. *A World of Fiction: Digital Collections and the Future of Literary History*. Ann Arbor: University of Michigan Press.

Daston, Lorraine, and Peter Galison. 2017. *Objectivity*. New York: Zone Books.

Drucker, Johanna. 2011. "Humanities Approaches to Graphical Display." *Digital Humanities Quarterly* 5, no. 1. http://www.digitalhumanities.org/dhq/vol/5/1/000091/000091.html.

Fagan, Benjamin. 2016. "Chronicling White America." *American Periodicals: A Journal of History & Criticism* 26, no. 1: 10–13. https://muse.jhu.edu/article/613375/summary.

Klein, Lauren. 2015. "The Carework and Codework of the Digital Humanities." http://lklein.com/2015/06/the-carework-and-codework-of-the-digital-humanities/.

Library of Congress. *Chronicling America* (newspaper archive). https://chroniclingamerica.loc.gov/.

Library of Congress. *U.S. Newspaper Directory*. https://chroniclingamerica.loc.gov/search/titles/.

Lupi, Giorgia, and Stefanie Posavec. 2016. *Dear Data: A Friendship in 52 Weeks of Postcards*. New York: Princeton Architectural Press.

Ramsay, Stephen. 2011a. *Reading Machines: Toward an Algorithmic Criticism*. Urbana: University of Illinois Press.

Ramsay, Stephen. 2011b. "Who's In and Who's Out." Position paper delivered at the 2011 Modern Language Association (MLA) Conference. https://web.archive.org/web/20170426170232/http://stephenramsay.us/text/2011/01/08/whos-in-and-whos-out/.

Schmidt, Benjamin M. 2016. "Do Digital Humanists Need to Understand Algorithms?" In *Debates in the Digital Humanities*, edited by Lauren F. Klein and Matthew K. Gold. Minneapolis: University of Minnesota Press. http://dhdebates.gc.cuny.edu/debates/text/99.

Vee, Annette. 2017. *Coding Literacy: How Computer Programming Is Changing Writing*. Cambridge, MA: The MIT Press.

Witmore, Michael. 2012. "Text: A Massively Addressable Object" [2010]. In *Debates in the Digital Humanities*, edited by Matthew K. Gold. Minneapolis: University of Minnesota Press. http://dhdebates.gc.cuny.edu/debates/text/28.