Evidence Based Library and Information Practice

Research Article

A Comparative Analysis of the Use of GitHub by Librarians and Non-Librarians

Mark E. Eaton
Assistant Professor / Reader Services Librarian
Kingsborough Community College
City University of New York
Brooklyn, New York, United States of America
Email: mark.eaton@kbcc.cuny.edu

Received: 12 June 2017 **Accepted**: 6 Apr. 2018

DOI: 10.18438/eblip29291

Abstract

Objective – GitHub is a popular tool that allows software developers to collaborate and share their code on the web. Librarians have adopted GitHub to support their own work, sharing code in support of their libraries. This paper asks: How does librarians' use of GitHub compare to that of other users?

Methods – To retrieve quantitative data on GitHub users, we queried the GitHub APIs (application programming interfaces). By assembling data on librarians' use of GitHub, as well as on a comparison group, we provided preliminary comparisons of these two samples. We analyzed and visualized this data across a number of variables to offer salient insights as to how librarians compare to randomly selected GitHub users.

Results – Librarians regularly use a more diverse range of programming languages than the comparison group, hinting at a broad range of possible uses of code in libraries. While the librarians' sample group did not demonstrate statistically significant differences from the comparison group on most measures of activity and popularity, they scored significantly higher in reach and productivity than the comparison group. This could be due to librarians' greater

longevity on GitHub, as well as their greater investment in GitHub as a tool for sharing.

Conclusion – Our data suggest that librarians are actively building their libraries with code and sharing the results. While it was unclear whether librarians were more active or popular on GitHub than the comparison group, it was clear that they demonstrated statistically significant outperformance in terms of reach and productivity. To explain these findings, we hypothesized that librarians' embrace of GitHub is in line with widely held values of "openness" in the library profession.

GitHub is "the biggest revelation in my workflow … since I started writing code." (Falster, as qtd. in Perkel, 2016, p. 127)

Introduction

GitHub is a well-known web-based code sharing platform that has recently exploded in popularity. Its functionality underpins cooperation by developers on many software projects by allowing programmers to share and promote their work. For scholars, GitHub is an interesting space to examine web collaboration and cooperative coding.

GitHub is built on git, a prominent software version control system that allows many geographically dispersed contributors to collaborate on a project asynchronously. Git is a command line tool that, among other things, allows for versioning, branching, and merging of projects' histories. GitHub adds value to git by providing a web interface to a great deal of git's functionality, and by adding additional social and workflow features, thereby making git-based projects much more accessible to the public and to other programmers.

Librarians who code presumably benefit from GitHub in many of the same ways as other programmers. It makes librarians' programming work accessible and open to the public, and it improves workflows via robust web tools. We can postulate that GitHub is an effective tool for collaboration amongst coding librarians in much the same way as it is for other users.

Aims

The goal of this paper is to answer the following question: How does librarians' use of GitHub compare to that of other users?

This question leads to more specific inquiries about librarians' use of GitHub. Are librarians more or less active on GitHub than other users? Are they more or less popular? Are they prolific producers of code? Are they more or less connected with other developers on the site? Do they tend to use the same programming languages as the larger community, or is their code clustered idiosyncratically in specific languages? Are their repositories well regarded?

We can begin to work toward answers to these questions using data from the GitHub APIs, or application programming interfaces. This paper describes how we queried the GitHub APIs to assemble quantitative data about GitHub use. We describe how we handled the data to make useful visualizations. Our analyses offer preliminary conclusions focused on our research question.

Our aim is to provide evidence-based insights about how librarians use GitHub. This is important, as librarians' software development work arguably plays a key role in the future of the profession. This paper makes a small step toward providing these important insights.

This paper does not explain how to use git, GitHub, or the GitHub APIs. Some understanding of GitHub and APIs is presumed.

For those looking for an introduction to git, Blischak, Davenport, and Wilson (2016), Perez-Riverol et al. (2016), and Bell and Beer (2015) have provided detailed how-tos. There are also plenty of interactive tutorials online that teach how to use git and GitHub, such as Lord (2014), Lord (2015), and GitHub (2017a). Others resources provide detailed introductions on how to use the GitHub APIs (Dataquest, 2017; GitHub, 2017b). This paper does not review these topics.

Literature Review

Scholarly literature on the use of GitHub in librarianship is almost nonexistent. If we broaden our focus beyond GitHub, we find that the scholarly literature on librarians' coding practices is only slightly less sparse. Townsdin and Whitmer (2016) describe it as a "limited literature" (p. 251). One might assume that this is because there are not many librarians writing code. However, the existence of prominent librarian professional associations that focus on technology skills—such as Code4Lib and the Library and Information Technology Association (LITA)—suggests otherwise.

Unfortunately, this "limited literature" does not offer many insights about why librarians are writing and using code. Some authors have speculated. Marshall (2015) suggests that librarians may use code "to solve problems, to improve or enhance existing applications, or to create new ones to meet specialized needs" (p. 25). Townsdin and Whitmer (2016) describe several use cases, including facilitating librarians' collaboration with IT departments, building websites, and wrangling data. Yelton (2015) suggests that "data import, export, and cleanup; expanded reporting capability; and patron facing services" (p. 2) are possible reasons to write code in libraries. Others suggest new, underexplored opportunities for code in librarianship, such as organizing hackathons (Davis, 2016). Yet these suggestions do not exhaust the possibilities. Somewhat more cynically, Stuart (2011) describes programming as an "obligation" (p. 43) for contemporary librarians; Townsdin and Whitmer (2016) imply the same when they point out that programming skills are often requirements of candidates looking for library jobs.

Some of the best insights into why librarians use code can be gleaned from Andromeda Yelton's (2015) extensive report on learning to code for librarians. Her interviews with over 50 librarians who have written code for their work give us important perspectives on librarians' coding practices. Upon reading through her examples, we get a sense of the preoccupations and interests of librarians who are either learning to program or who are actively using code. Likewise, Enis (2013) helpfully provides real world examples of individual librarians' coding practices. While the samples of librarians assembled by Yelton and Enis do not appear to be controlled or random, they do provide useful and interesting narrative examples of why librarians might use code.

Apart from librarianship, scholarship on GitHub is a new and growing field. In 2015, Grier argued that "scholars have drawn only modest conclusions from their study of the Github database" (p. 116). Longo and Kelley identified a similar lack of scholarship in 2016. However, the study of GitHub is changing quickly. Grier's description may have been accurate at the time he was writing; however, git and GitHub scholarship have developed substantially since then.

Much of the existing work on GitHub is concentrated in the field of computer science. Given that git and GitHub are often taken for granted among professional software developers, it is perhaps unsurprising that computer science is well ahead of other disciplines in producing scholarship on GitHub. There are many examples of this type of literature: Arora, Goel, and Mittal (2016) analyze syntactic and semantic conflicts that arise when code is being written by multiple developers; Blincoe, Sheoran, Goggins, Petakovic, and

Damian (2016) measure the level of influence of popular users on GitHub; Hu, Zhang, Bai, Yu, and Yang (2016) also attempt to measure influence; Jiang et al. (2017) look at forking practices; Kalliamvakou et al. (2016) examine the quality of data that can be gathered from the GitHub APIs; Lima, Rossi, and Musolesi (2014) parse interactions on GitHub to analyze social activity; McDonald, Blincoe, Petakovic, and Goggins (2014) consider factors associated with successful open source projects on GitHub; and Yan, Wei, Han, and Wang (2017) analyze the use of GitHub for blogging.

It is clear that GitHub is important to software developers. However, Davis (2015) argues that these tools are also useful and important to librarians. While GitHub was not designed explicitly for librarians, it is a tool that has increasingly proved useful in many disciplines, including librarianship. Longo and Kelley (2016) argue that GitHub's popularity has spread to a range of diverse fields. As GitHub spreads across disciplinary boundaries, scholarship on GitHub begins to appear in many fields.

In other words, we are not alone in attempting to broaden the scholarly conversation on GitHub beyond computer science. The current growth of scholarly literature on GitHub mirrors the ongoing growth and expansion of GitHub's user base. While git has existed since 2005, the subsequent growth of GitHub has fueled the expansion of git-based workflows beyond git's core developer community: "As the uses and users of GitHub move beyond its original core community of software developers, the present and potential impact on fields such as social knowledge creation, open science, open collaboration, and open governance warrants consideration" (Longo & Kelley, 2016, p. 617). Recent work has been done on the use of GitHub in fields as diverse as computational biology (Blischak et al., 2016; Perez-Riverol et al., 2016) and public administration (Longo & Kelley, 2016; Mergel, 2015), among others.

While much of this scholarship is fairly recent, version control and code sharing did not begin with GitHub. Christopher Kelty (2008) demonstrates that a culture of sharing code has a very long history in computing. A number of version control tools have been used in open source projects since the early 1990s, and many of these were popular in their respective day. However, unlike GitHub, many of these early tools were only suitable for small development teams (Hu et al., 2016). Moreover, they have largely waned in popularity recently as git and GitHub have become ascendant.

Our methodological approach—gathering quantitative data from the GitHub API—is not entirely unique. Other scholars have done API-based work on GitHub, including some large-scale data gathering by Jiang et al. (2017); as well some interesting work by authors who have supplemented their API-based work with qualitative surveys (Blincoe et al., 2016; Mergel, 2015); and in several interesting projects presented at conferences, described by Hu et al. (2016).

Beyond these nearest methodological cousins, others do similar work while opting not to use the GitHub API. These latter studies instead rely on third party data-gathering tools or collections to access GitHub data. For example, McDonald et al. (2014) draw from the API using a tool called GitMiner; Lima et al. (2014) do similar data mining using a tool called The GitHub Archive; and a number of projects use GHTorrent, which is a mirror of the GitHub API (Blincoe, 2016; Kalliamvakou et al., 2016; Miller, 2016). While these studies are interesting and inform our present work, to our knowledge there has not been a study that directly examines the use of GitHub by librarians with quantitative methods. This study was intended to fill this gap by offering some preliminary quantitative analysis.

Method

This study compares two distinct samples of GitHub users across a number of variables, namely: *programming language choice*, *number of followers*, *number of following*, *number of public repositories, number of repository stars, GH index*, and *account creation date*. We have gathered data related to these variables from the GitHub API.

Our methodology focused in part on exploratory data visualization. Many scholars who have studied GitHub have made a point of presenting their findings visually (Blincoe et al., 2016; Hu et al., 2016; Jiang et al., 2017; Kalliamvakou et al., 2016; Lima et al., 2014; McDonald et al., 2014; Mergel, 2015; Yan, Wei, Han, and Wang, 2017). While there is a longstanding tradition of exploratory data visualization in scholarly literatures, stretching at least as far back at John Tukey (1977) and Edward Tufte (1983), a robust popular visualization literature has only developed more recently. This latter body of work is often aimed at professional practitioners rather than scholars (Cairo, 2013; Few, 2012; Knaflic, 2015; Wong, 2010; Yau, 2011, 2013). This paper draws on the insights of both of these traditions, insofar as they have taught us to think rigorously about effective visualization and to use data as an exploratory tool.

In his foundational text *Exploratory Data Analysis*, Tukey (1977) distinguishes between "exploratory" and "confirmatory" data analysis (p. 3). Subsequently, he argues that exploratory data analysis is necessary to successfully implement a confirmatory analysis (1980). In his later work with Hoaglin et al. (1983), he goes on to say that "exploratory data analysis emphasizes flexible searching for clues and evidence, whereas confirmatory data analysis stresses evaluating the availability of evidence" (p. 2). Tukey's encouragements toward exploratory analyses are taken up much later in the popular and technical literature on data visualization. For example, Nathan Yau (2011, 2013) follows in Tukey's footsteps in his work on data visualization.

Exploratory data analysis, in the tradition of Tukey and Yau, provided us with a methodological starting point. We also supplemented this with confirmatory analysis in the form of t-tests and chi-square tests to ensure that our exploratory work was on the right track and to evaluate the significance of our findings.

We gathered data from the GitHub APIs, and then manually and programmatically filtered the data to produce a useful, workable dataset. We then applied scientific Python tools to our data. Python is a popular programming language with well-regarded data science libraries. Our work used the *numpy*, *scipy* and *pandas* libraries to structure our data and run statistical tests, *matplotlib* and *seaborn* to plot charts, and other very common Python libraries like *requests* and the Python standard library. We made the code we wrote publicly available on GitHub (Eaton, 2016b).

We wrote code that retrieves and processes data from the GitHub APIs. Understanding how APIs work is helpful in understanding the methodology described below. While many APIs have a very wide range of possible use cases, our approach ignored most of these, and instead focused on how we could harvest quantitative data from the GitHub APIs. Below is a brief summary of how the code we wrote assembled data from GitHub.

1. We began by using the GitHub Search API to identify librarians on GitHub. We did this to establish the primary sample of subjects to study. We queried the Search API for the terms "library," "librarian," "libraries," "bibliothèque," and "bibliothecaire." (We included the French terms to increase our sample size.) To see an example of captured Search data, please consult Appendix A, which shows the JSON for an individual record as it appears when retrieved from the Search API. JSON, or JavaScript Object Notation, is a very

common data serialization format (JSON, 2016).

A unique request was constructed for each of the search terms mentioned above. Overall, this approach worked very well. However, the GitHub Search API returns a maximum of 100 results per query. Therefore, if there were more than 100 results, our initial API call would not retrieve all of the available data. For this reason, it was preferable not to use stemming. Instead, we searched each term individually because this maximized the number of results we retrieved.

We were able to partly work around the 100-result limitation by applying both ascending and descending sort order to the resulting data. This technique allowed us to get at both ends of the search results, meaning that we could capture up to 200 results per search, rather than just 100. This was sufficient to retrieve all of the results for the keywords "librarian," "biblothèque," and "bibliothecaire." However, the searches for "library" and "libraries" yielded more than 200 hits. For these larger results sets, the total number of results was in the low four figures. As a result, in these cases, our script only captured the first 100 and the last 100 results. This situation was not ideal, but we decided this was acceptable for our purposes, since our searches provided us with sufficient data to conduct statistically significant analysis.

The data produced by these searches was deduplicated, concatenated, and saved as a JSON document.

2. Alongside the list of librarians and libraries that were generated, we also wrote a function that generated a random comparison group of GitHub users. This did not use the Search API, but rather generated random integers that could be passed to the User API as user ID numbers. At the time of our analysis, the range of valid GitHub user IDs went up to around 20,000,000. Our method involved randomly generating user IDs to create a randomized sample of non-librarians.

3. To maintain data quality, it was also important for us to distinguish between libraries in the traditional sense—as institutions, physical spaces, and collections—from libraries in the programming sense—as software packages. For our *librarians* sample, we were only interested in the former. For our *comparison group* sample, we were interested in excluding librarians. We considered using a natural language processing approach to make these distinctions; however, we ultimately decided that our dataset was small enough to sort through our results manually. This process allowed us to weed out software libraries from the *librarians* dataset. Our manual approach allowed us to maintain a high degree of accuracy.

4. We also needed to ensure that no librarians were placed in the *comparison group* by the function that generates the randomly selected *comparison* users. For this reason, we also manually reviewed each record in the *comparison group* to confirm that they were not librarians. We are confident that our manual verification of both the *librarians group* and *comparison group* was effective; nonetheless, any potential error introduced by this process is a limitation of this study.

5. For both groups we retrieved data on each individual user from the GitHub User API. This returned JSON data

about that user. We captured all of the user data provided by the API; however, we used only a small subset of that data in our subsequent analysis. To see an example of captured user data, please consult Appendix B, which shows User API JSON for an individual user.

6. We then retrieved repository-level data for both groups of users, using the GitHub Repo API. This returned JSON that describes that user's repositories. Again, we captured all of the repository data provided by the API; however, we used only a small subset of that data in our subsequent analysis. To see an example of captured repository data, please consult Appendix C, which shows some Repository API JSON for an individual user.

7. To improve the quality of the data ultimately used for analysis, it was necessary to filter the data about our two sample groups across a number of criteria. There are numerous reasons to

Table 1
Criteria for Selection and Their Justification

Our criteria	**Reasons for the criteria**
The user must have been active during the last 90 days.	We wanted a contemporary picture of GitHub, so our focus was on current users. To this end, we excluded users whose *updated_at* date was more than 90 days old.
The user account must be more than 30 days old.	We deliberately excluded very new sign-ups to focus on those users who had an established presence on GitHub. We did this by excluding users whose *created_at* date was less than 30 days old.
The user must have contributed to at least one public repository. It is important to note that this contribution can be to someone else's repository; in other words, it is not necessarily their own repository.	A significant number of users sign up for a GitHub account but contribute nothing. These abandoned accounts would have produced almost no interesting data and would have crowded out accounts that have data that is interesting and useful. For these reasons, we excluded these users.
The user must have a bio.	Because querying the Search API for keywords (such as "librarian") favours those profiles that have bios, we required that all users included in the study have bios. This made the *comparison group* and the *librarians group* more directly comparable.
The list of users must be deduplicated.	For obvious reasons we did not want to count the same user twice.
The *librarians* dataset and the *comparison* dataset need to be the same size.	This was done to make for an easy comparison of the two groups being studied.

carefully filter GitHub data, some of which are described by Kalliamvakou et al. (2016). Many of the techniques they describe for mitigating the "perils" of using GitHub data are reflected in the approaches to filtering data that we used in our scripts. The filtering techniques that we used are described in Table 1.

Once we had gathered and processed our data, the final samples used in our analyses consisted of 112 *librarian*s and 112 *comparison group* subjects.

Results and Discussion

The most obvious place for us to begin our analysis was to compare the programming languages used by *librarians* to those used by the *comparison group* of GitHub users. Figures 1 and 2 show the top 15 languages for both groups. In total, there were 1,433 repositories for *librarians* (average of 12.79 per *librarian*) and 1,075 repositories for the *comparison group* (average of 9.60 per *comparison group* user). The same scale is used in Figures 1 and 2 for easy comparison. A chi-square test was run on the top 12 languages that are common to both *librarians* and the *control group*. The most striking aspect of the results is the highly significant difference between the two groups ($X^2(11, N = 1840) = 282.70$; $p < .001$). The *p-value* for this chi-square test is < .001, far below the conventional threshold for statistical significance of $p < .050$.

The *comparison group*'s language choices are concentrated specifically in JavaScript and Java, while the *librarians*' language choices are more evenly distributed over a wider range of programming languages. The *librarians*' wide variety of language choices may reflect the many different possible uses of code in libraries. Libraries are home to many diverse activities, possibly resulting in many varied reasons for adopting different programming languages.

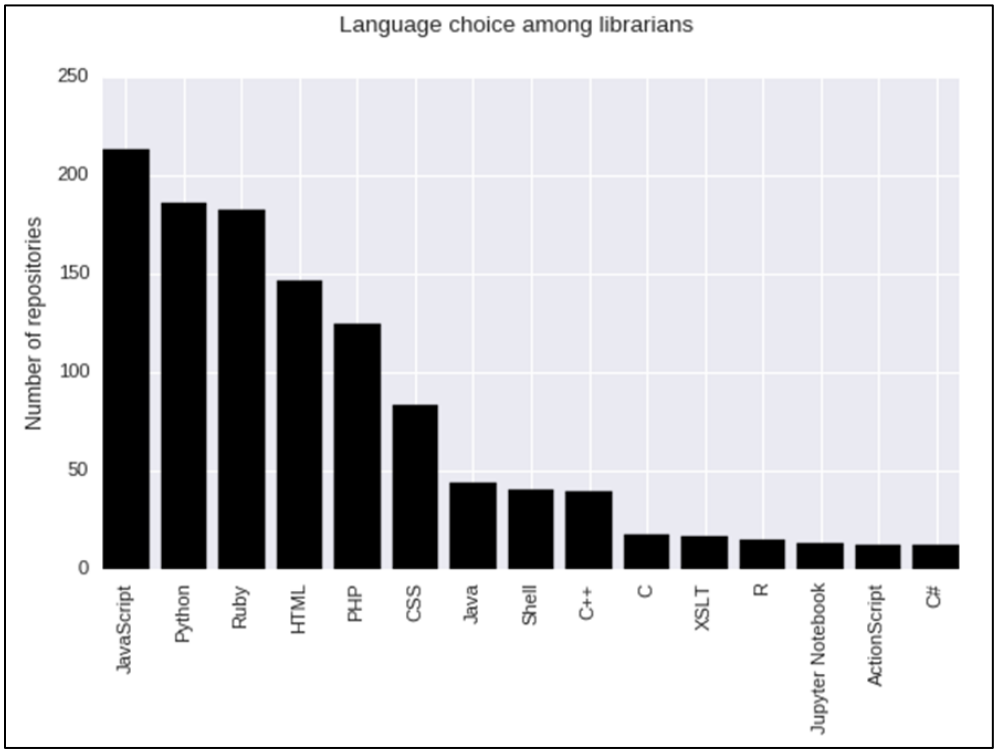

Figure 1
Language choice among librarians

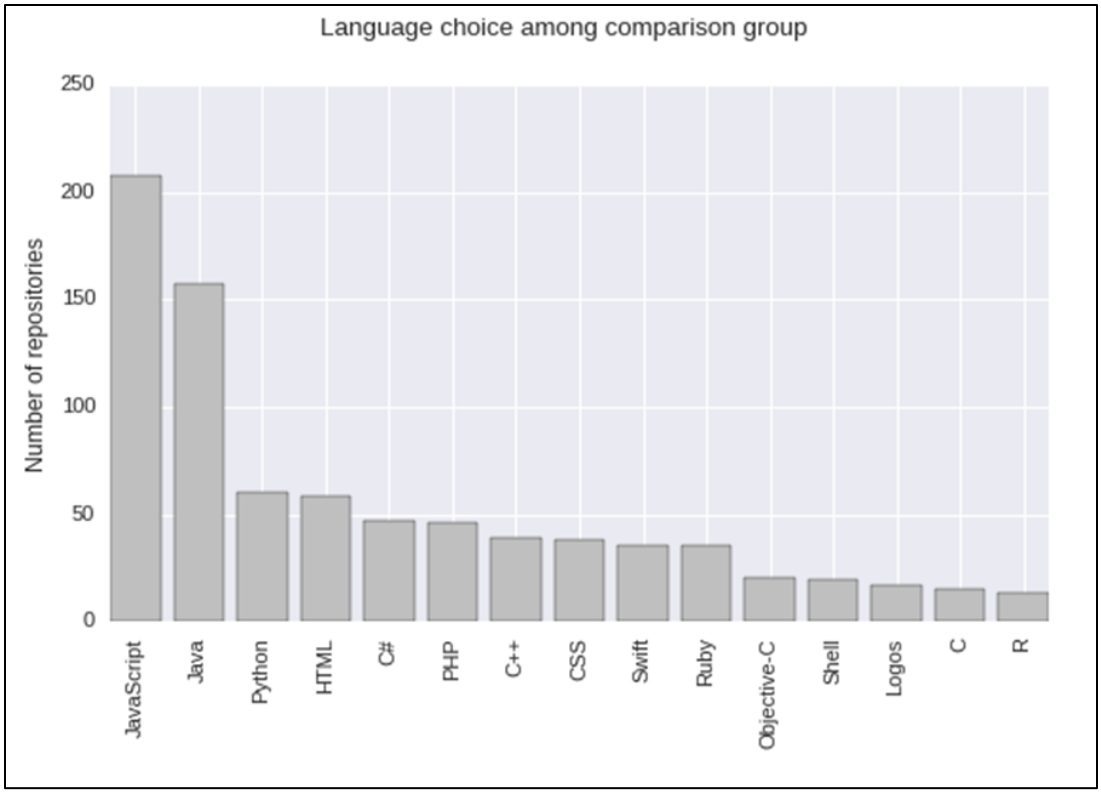

Figure 2
Language choice among comparison group

Interestingly, the most popular languages differ as well. JavaScript, the preeminent language of the web, perhaps unsurprisingly dominates both groups. But from there the two samples diverge. While Java features so prominently in the *comparison group*'s preferred choices, it comes in at a distant seventh among *librarians*. Ruby is strongly favoured by *librarians*, while C# is strongly favoured by the *comparsion group*. While we will forgo discussing the merits of various languages, it is clear that librarians are choosing noticeably different languages for their projects. What we learned from these charts is that librarians are using a broad range of programming options, suggesting a range of possible use cases for code in libraries.

Next, we turned our attention to user metrics such as *number of followers*, *number of following*, and *number of public repositories* for users in both groups. "Followers" and "following" should be familiar concepts to users of most social media. In the GitHub context, if I "follow" someone, the result is that their activity (such as creating, forking, or starring repositories) will appear in a timeline on my GitHub home screen. *Public repositories* are projects that a user has shared on GitHub. A repository can be either created from scratch by a user or derived from another user's work ("forked," in GitHub terms).

We applied two-tailed t-tests to these variables for confirmatory purposes. Because of the very high variance of the data, we ran a log transformation on the original data before doing the t-tests on the transformed data. From these tests, we found that there is not a statistically significant difference in terms of *number of followers* ($t(222) = 1.62$, $p = .107$); *number of following* ($t(222) = 0.91$, $p = .363$); and *number of public repositories* ($t(222) = 1.52$, $p = .131$) between the two groups.

Stars are another GitHub concept that should be familiar to users of other social media. However, *stars* are used somewhat differently in GitHub than in other social media. GitHub users star repositories, rather than individual messages or posts. In this respect, starring is more of an endorsement of a project, rather than a reaction to a specific message or post from another user.

Interestingly, there is a statistically significant difference in the *number of repository stars* ($t(222) = 2.00$, $p = .048$) for the two sample groups. *Librarians* have significantly more repository stars than the *comparison group*, according to a two-tailed t-test. Because of the high variance in the data, we first ran a log transformation on the original *number of stars* data, and then did our t-test on the transformed data.

To summarize, the results of t-tests thus far can be seen in Table 2.

To use a more exploratory, data visualization approach—in the spirit of Tukey (1977) and Yau (2011, 2013) —it was also interesting to visually group these datasets into those that measure activity (*number of public repositories* and *number of following*) and those that measure popularity (*number of repository stars* and *number of followers*) via scatterplots. This approach produced the charts in Figure 3 and Figure 4, respectively.

We used a log scale for these plots for increased readability and for consistency with our confirmatory analysis. Visually, a log scale accommodates outlying users who were disproportionately more active or more popular than most of the subjects. Also, we added one (1) to each value, so that all values would be displayed on the scatterplots, as a log scale cannot display zero values. It should be noted that this causes some distortion in the lower left-hand corner of the scatterplots.

To facilitate comparison of the two samples in aggregate, we included the averages for both groups, displayed on the scatterplots as a star. While the *number of repository stars* is the only statistically significant variable in these calculations, it is interesting to note that the mean for librarians tended to be higher than the mean for the comparison group for all of the other variables as well. This relationship is especially pronounced in the popularity scatterplot and can be seen in Table 2 as well.

Given the high variance found in all of these variables, it would be useful to conduct further analysis with a larger study sample to demonstrate (or alternately disprove) a statistically significant relationship between these factors. In this way, our exploratory visual analysis suggests directions for future, larger-scale confirmatory analyses.

Table 2
Summary of Results from T-Tests

Variable	**P-value of log(Variable)**	**Mean, librarians**	**Mean, comparison group**
Number of followers	.107	107.19	12.61
Number of following	.363	13.24	8.85
Number of public repositories	.180	19.05	15.47
Number of repository stars	.048	41.96	9.0

P-value threshold used for statistical significance: < .050.

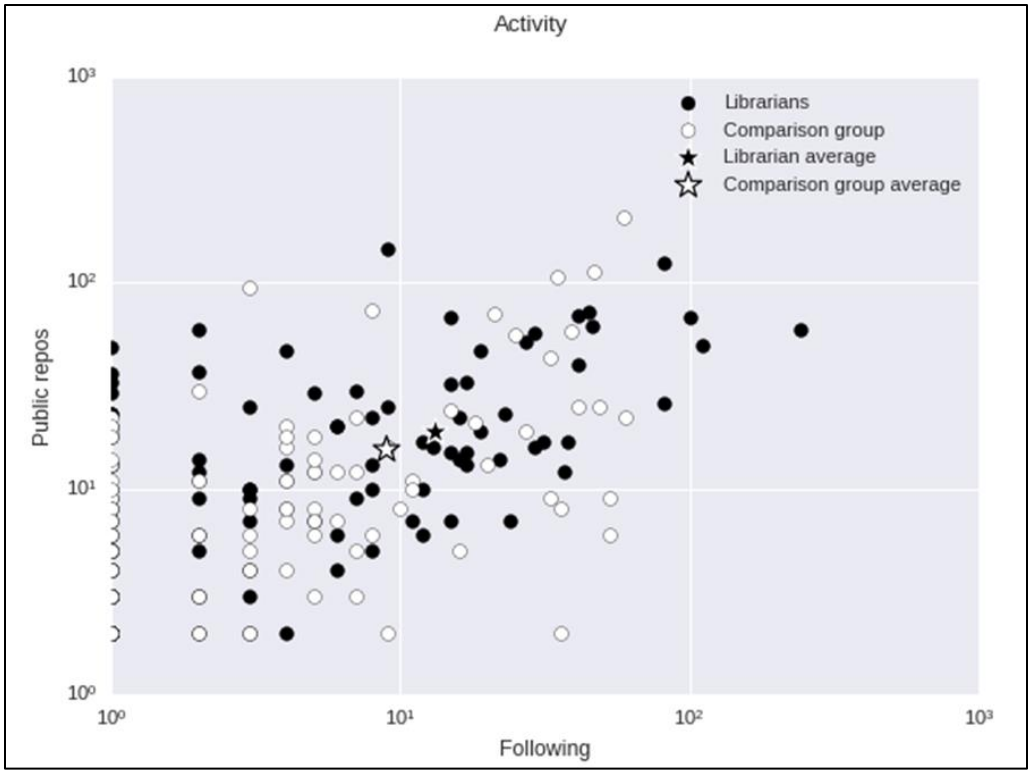

Figure 3
Activity

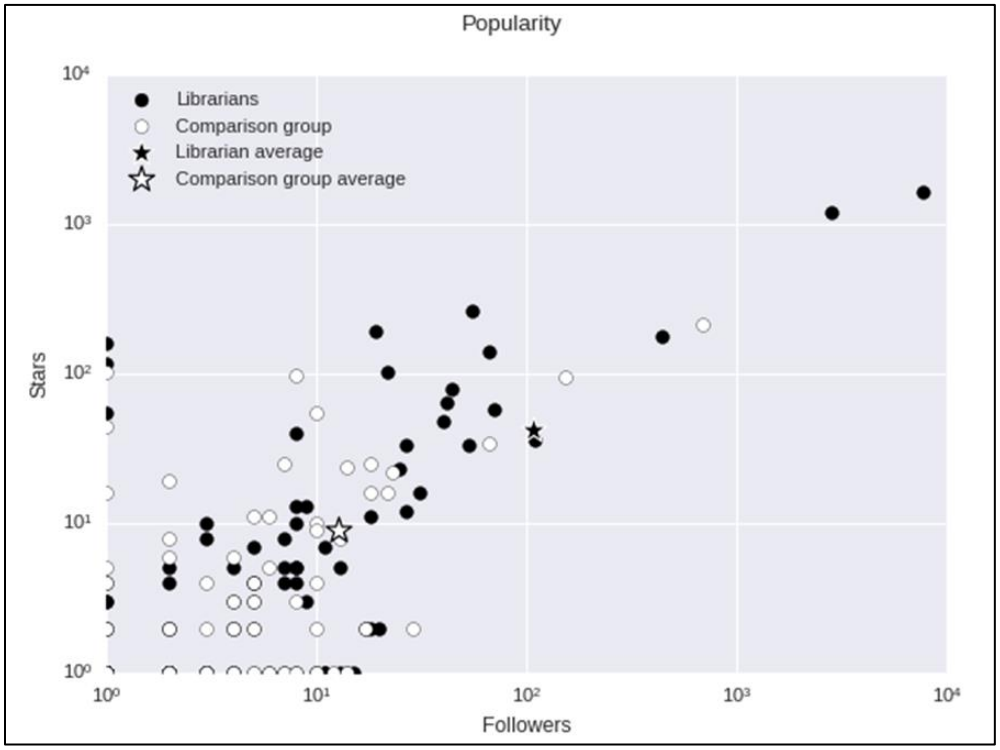

Figure 4
Popularity

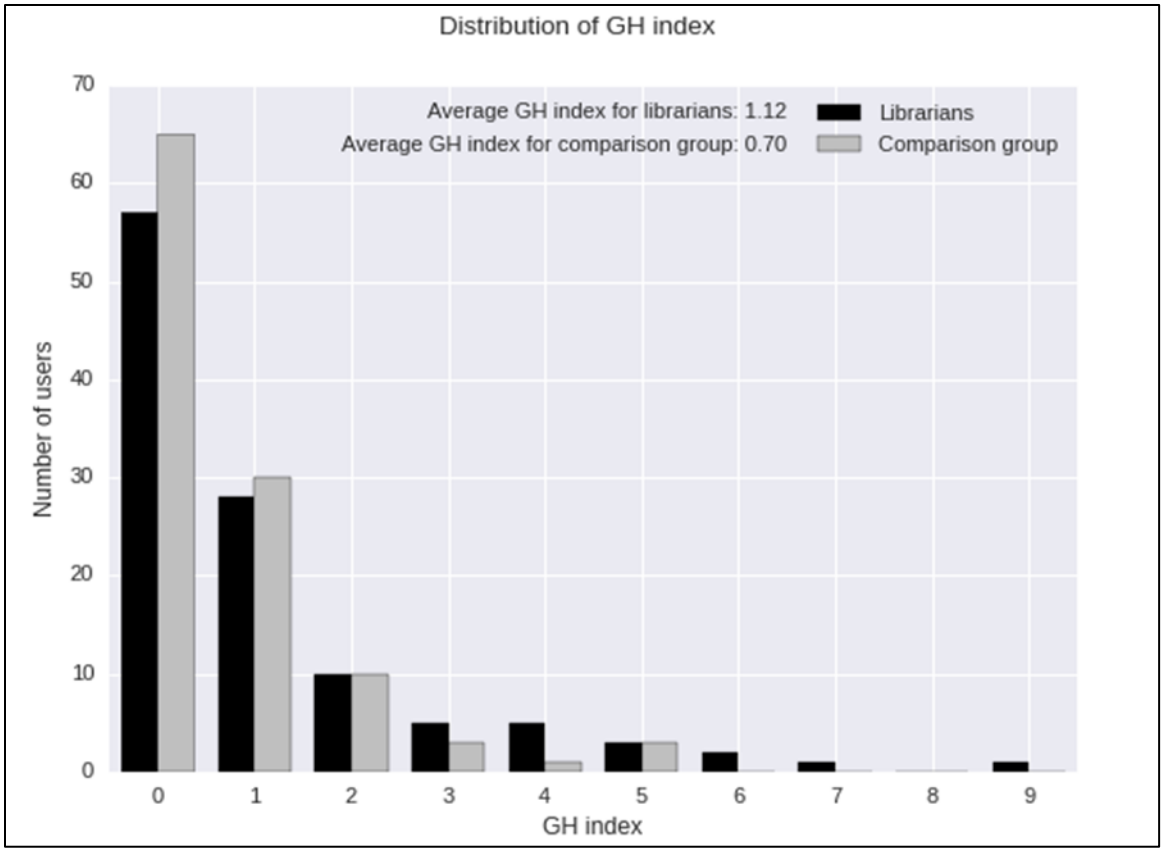

Figure 5
Distribution of GH index

It is also interesting to apply a more sophisticated measure to evaluate the *number of repository stars*. Elsewhere, I have devised a measure called GH index, which measures the reach and productivity of GitHub users (Eaton, 2016a). GH index uses the same math as the widely known H-index measure, which measures the reach and productivity of a scholar. The innovation of GH Index is that it applies the logic of H-index to GitHub stars rather than academic citations. To our knowledge, GH Index is a novel measure of contributions to open source projects. Miller (2016) later adopted this same measure and further popularized it, naming it GH Impact.

When we chart our subject groups according to *GH index* score, we get the histogram shown in Figure 5.

While the two groups are fairly similar according to *GH index*, librarians do have an edge, implying greater productivity and greater reach for their projects. We applied a t-test to this relationship, as statistical confirmation. Because of much lower variance of the *GH index* data, we did not apply a log transformation when doing this particular test. Thus, we can see that *librarians* (M = 1.12) score significantly higher than the *comparison group* (M = 0.70) in productivity and reach ($t(222) = 2.22$, $p = .027$), which is statistically significant at the conventional $p < .050$ level.

Considering our findings on *number of repository stars* and *GH index*, we could potentially argue, following Hu et al. (2016), that "popularity and quality [of GitHub repositories] are strong indicators of their owners' capability" (p. 5). However, because Hu et al. do not adequately

support this claim, we are reluctant to state the case as strongly as they do. Nonetheless, our measures of *repository stars* and *GH index* lead us to suggest that our *librarians*' repositories are making a greater impact than those of our *comparison group*.

This exploratory and confirmatory analysis of librarians' popularity, activity, productivity, and reach leads us to ask a more specific follow-up question: What is librarians' level of influence on GitHub? Blincoe et al. (2016) have measured the effect of various metrics on the level of influence that a user has on GitHub. They point out "that popular users often attract their followers to new projects." Moreover, they argue that "users who are both very popular and very active influence their followers" (p. 31). If Blincoe et al. are correct about this, popular *librarians* might similarly benefit from the added influence of their GitHub reputation. This is an interesting suggestion; however, two issues prevent us from generalizing Blincoe et al.'s conclusions to our dataset. First, only two librarians in our study reach the popularity threshold of 500 followers that Blincoe et al. require to be included in their analysis of influence. Second, while our exploratory analysis suggests that librarians may possibly be more active and popular than our comparison sample, this is not confirmed by t-tests. Our statistically significant variables, *number of repository stars* and *GH index*, are not sufficient for an analysis along the lines of Blincoe et al. Because of these factors, Blincoe's conclusions are not generalizable to our dataset.

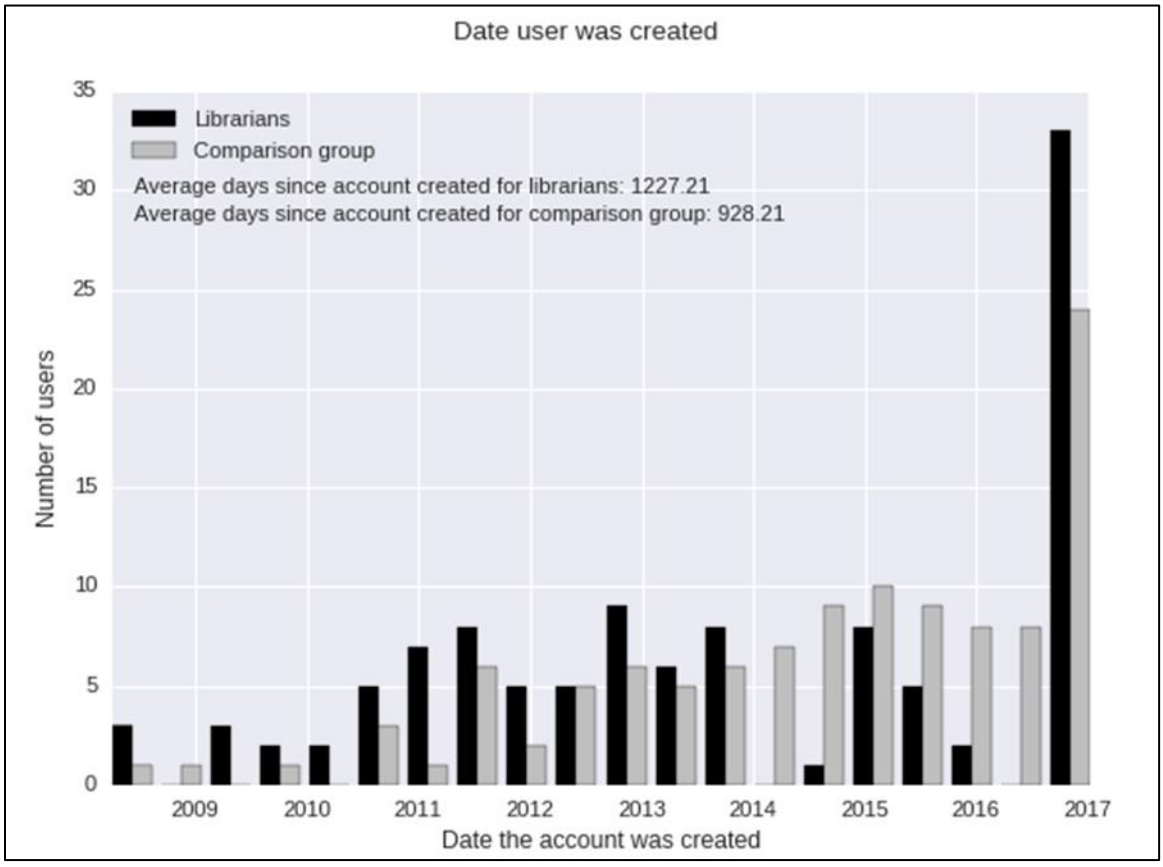

Figure 6
Account creation date

One factor that may help to explain our findings is that librarians can be shown to be early adopters of GitHub. Specifically, if librarians have been on GitHub longer than the average user, it can explain their relative productivity and reach, as shown in the data on *repository stars* and *GH index*. It is therefore interesting to plot a histogram that shows the *account creation date* of our samples' user accounts, showing how long they have been on GitHub (see Figure 6). This relationship, when tested with a two-tailed t-test, yields a highly significant *p-value* ($t(222) = 2.64$, $p = .009$). We can therefore confidently say that *librarians* (M = 1227.21 days) tend to have been on GitHub longer than our *comparison group* users (M = 928.21 days).

These analyses provide many interesting insights to consider. We hypothesize that librarians' measurable and statistically significant involvement with GitHub is the result of their profession's embrace of GitHub as an "open" platform. Sharing code publicly often presupposes a certain commitment to openness. Oftentimes this openness is a legal category, assigned by the programmer or the institution as a software license. Kelty (2008) describes the history of openness in software communities. Openness may also be a powerful motivator for some librarians who share a commitment to the value of openly sharing information in all formats (Puckett, 2012; Fernandez, 2012). GitHub provides a way for librarians to put their commitments to openness into action by providing a highly visible way to share code freely.

Conclusion

Code sharing is an important topic in librarianship because librarians shape their libraries and communities through the software they build. By programming for libraries, librarians are directly contributing to what their libraries will be in the future. Librarians' work builds their institutions with code. In this sense, librarians create software tools that produce "actually existing alternatives" (Kelty, 2008, p. 3) for libraries.

We have preliminarily established that the *librarians* in our study demonstrate statistically significant outperformance in reach and productivity on GitHub. They also closely mirror the *comparison group* on measures of activity and popularity. We have pointed out that librarians use diverse programming languages, perhaps as a result of their diverse librarianship practices. Moreover, we hypothesize that librarians' embrace of GitHub is rooted in values of openness. Hopefully this study has demonstrated that librarians are indeed significant users of GitHub and that further confirmatory study of these topics is warranted.

References

Arora, R., Goel, S., & Mittal, R. K. (2016). Supporting collaborative software development over GitHub. *Software: Practice and Experience*, *47*, 1393–1416. https://doi.org/10.1002/spe.2468

Bell, P., & Beer, B. (2015). *Introducing GitHub: A non-technical guide*. Beijing: O'Reilly.

Blincoe, K., Sheoran, J., Goggins, S., Petakovic, E., & Damian, D. (2016). Understanding the popular users: Following, affiliation influence and leadership on GitHub. *Information and Software Technology*, *70*, 30–39. https://doi.org/10.1016/j.infsof.2015.10.002

Blischak, J. D., Davenport, E. R., & Wilson, G. (2016). A quick introduction to version control with git and GitHub. *PLoS Computational Biology*, *12*(1). https://doi.org/10.1371/journal.pcbi.1004668

Breeding, M. (2015). More than a hack: Empowering your library through

coding. *Computers in Libraries*, *35*(6), 25–27.

Cairo, A. (2013). *The functional art: An introduction to information graphics and visualization*. Berkeley, CA: New Riders.

Dataquest. (2015, September 8). Python API tutorial: An introduction to using APIs. Retrieved May 24, 2017, from https://www.dataquest.io/blog/python-api-tutorial/

Davis, R. C. (2015). Git and GitHub for librarians. *Behavioral & Social Sciences Librarian*, *34*(3), 159–164. https://doi.org/10.1080/01639269.2015.1062586

Davis, R. C. (2016). Hackathons for libraries and librarians. *Behavioral & Social Sciences Librarian*, *35*(2), 87–91. https://doi.org/10.1080/01639269.2016.1208561

Eaton, M. (2016a). I made my own altmetric. Retrieved April 12, 2017, from https://kingsboroughlibtech.commons.gc.cuny.edu/2016/06/14/i-made-up-an-altmetric/

Eaton, M. (2016b). GitHub-Study. *GitHub.* Retrieved April 12, 2017, from https://github.com/MarkEEaton/github-study

Enis, M. (2013). Cracking the code. *Library Journal*, *138*(4), 24.

Fernandez, P. (2012). Library values that interface with technology: Public service information professionals, Zotero and open source decision making. *Library Philosophy and Practice*, 1–11. https://digitalcommons.unl.edu/libphilprac/803/

Few, S. (2012). *Show me the numbers: Designing tables and graphs to enlighten* (2nd ed.). Burlingame, CA: Analytics Press.

GitHub (2017a). Git and GitHub learning resources: User documentation. Retrieved April 10, 2017, from https://help.github.com/articles/git-and-github-learning-resources/

GitHub (2017b). GitHub API v3: GitHub developer guide. Retrieved April 11, 2017, from https://developer.github.com/v3/

Grier, D. A. (2015). The GitHub effect. *Computer*, *48*(5), 116. https://doi.org/10.1109/MC.2015.146

Hoaglin, D. C., Mosteller, F., & Tukey, J. W. (1983). *Understanding robust and exploratory data analysis*. New York: John Wiley & Sons.

Hu, Y., Zhang, J., Bai, X., Yu, S., & Yang, Z. (2016). Influence analysis of GitHub repositories. *SpringerPlus*, *5*, 1–19. https://doi.org/10.1186/s40064-016-2897-7

Jiang, J., Lo, D., He, J., Xia, X., Kochhar, P. S., & Zhang, L. (2017). Why and how developers fork what from whom in GitHub. *Empirical Software Engineering*, (22), 547–578. https://doi.org/10.1007/s10664-016-9436-6

JSON. (2016). Retrieved February 23, 2016, from http://www.json.org/

Kalliamvakou, E., Gousios, G., Blincoe, K., Singer, L., German, D. M., & Damian, D. (2016). An in-depth study of the promises and perils of mining GitHub. *Empirical Software Engineering*, *21*, 2035–2071. https://doi.org/10.1007/s10664-015-9393-5

Kelty, C. M. (2008). *Two bits: The cultural significance of free software*. Durham, NC: Duke UP.

Knaflic, C. N. (2015). *Storytelling with data: A data visualization guide for business professionals*. Hoboken, NJ: John Wiley and Sons, Inc.

Lima, A., Rossi, L., & Musolesi, M. (2014). Coding together at scale: GitHub as a collaborative social network. *arXiv* (1407.2535). https://arxiv.org/abs/1407.2535

Longo, J., & Kelley, T. M. (2016). GitHub use in public administration in Canada: Early experience with a new collaboration tool. *Canadian Public Administration*, *59*(4), 598–623. https://doi.org/10.1111/capa.12192

Lord, J. (2014). jlord/git-it. Retrieved April 10, 2017, from https://github.com/jlord/git-it

Lord, J. (2015). jlord/git-it-electron. Retrieved April 10, 2017, from https://github.com/jlord/git-it-electron

McDonald, N., Blincoe, K., Petakovic, E., & Goggins, S. (2014). Modeling distributed collaboration on GitHub. *Advances in Complex Systems*, *17*(7-8). https://doi.org/10.1142/S0219525914500246

Mergel, I. (2015). Open collaboration in the public sector: The case of social coding on GitHub. *Government Information Quarterly*, *32*, 464–472. https://doi.org/10.1016/j.giq.2015.09.004

Miller, I. D. (2016). GH-impact measures open source influence. Retrieved April 10, 2017, from http://www.gh-impact.com

Perez-Riverol, Y., Gatto, L., Wang, R., Sachsenberg, T., Uszkoreit, J., da Veiga Leprevost, F., Fufezan, C., Ternent, T., Eglen, S. J., … Vizcaino, J. A. (2016). Ten simple rules for taking advantage of git and GitHub. *PLoS Computational Biology*, *12*(7). https://doi.org/10.1371/journal.pcbi.1004947

Perkel, J. (2016). Democratic databases: Science on GitHub. *Nature*, *538*, 127–128.

Puckett, J. (2012). Open source software and librarian values. *Georgia Library Quarterly*, *49*(3), 30–34. https://scholarworks.gsu.edu/univ_lib_facpub/95/

Stuart, D. (2011). Programming librarians in the web of data. *Online*, *35*(2), 42–44.

Townsdin, S., & Whitmer, S. (2016). "Hello, World!": Starting a coding group for librarians. *Public Services Quarterly*, *12*(3), 249–256. https://doi.org/10.1080/15228959.2016.1197082

Tufte, E. R. (1983). *The visual display of quantitative information*. Cheshire, CT: Graphics Press.

Tukey, J. W. (1977). *Exploratory data analysis*. Reading, MA: Addison-Wesley.

Tukey, J. W. (1980). We need both exploratory and confirmatory. *The American Statistician*, *34*(1), 23–25. https://doi.org/10.1080/00031305.1980.10482706

Wong, D. M. (2010). *The Wall Street Journal guide to information graphics: The do's and don'ts of presenting data, facts, and figures*. New York: W.W. Norton & Co.

Yan, D.-C., Wei, Z.-W., Han, X.-P., & Wang, B.-H. (2017). Empirical analysis on the human dynamics of blogging behavior on GitHub. *Physica A*, *465*, 775–781. https://doi.org/10.1016/j.physa.2016.08.054

Yau, N. (2011). *Visualize this: The FlowingData guide to design, visualization and statistics*. Indianapolis, IN: Wiley.

Yau, N. (2013). *Data points: Visualization that means something*. Somerset, NJ: John Wiley & Sons.

Yelton, A. (2015). *Coding for Librarians: Learning by example* (Library Technology Reports No. 51(3)). Chicago, IL: American Library Association.

Appendix A
Data from the GitHub Search API


```
{
  "total_count": 322,
  "incomplete_results": false,
  "items": [
    {
      "login": "octocat",
      "id": 583231,
      "avatar_url": "https://avatars3.githubusercontent.com/u/583231?v=4",
      "gravatar_id": "",
      "url": "https://api.github.com/users/octocat",
      "html_url": "https://github.com/octocat",
      "followers_url": "https://api.github.com/users/octocat/followers",
      "following_url": "https://api.github.com/users/octocat/following{/other_user}",
      "gists_url": "https://api.github.com/users/octocat/gists{/gist_id}",
      "starred_url": "https://api.github.com/users/octocat/starred{/owner}{/repo}",
      "subscriptions_url": "https://api.github.com/users/octocat/subscriptions",
      "organizations_url": "https://api.github.com/users/octocat/orgs",
      "repos_url": "https://api.github.com/users/octocat/repos",
      "events_url": "https://api.github.com/users/octocat/events{/privacy}",
      "received_events_url": "https://api.github.com/users/octocat/received_events",
      "type": "User",
      "site_admin": false,
      "score": 114.60762
    },
    ...
  ]
}
```

Appendix B
Data from the GitHub User API


```
{
  "login": "octocat",
  "id": 583231,
  "avatar_url": "https://avatars3.githubusercontent.com/u/583231?v=4",
  "gravatar_id": "",
  "url": "https://api.github.com/users/octocat",
  "html_url": "https://github.com/octocat",
  "followers_url": "https://api.github.com/users/octocat/followers",
  "following_url": "https://api.github.com/users/octocat/following{/other_user}",
  "gists_url": "https://api.github.com/users/octocat/gists{/gist_id}",
  "starred_url": "https://api.github.com/users/octocat/starred{/owner}{/repo}",
  "subscriptions_url": "https://api.github.com/users/octocat/subscriptions",
  "organizations_url": "https://api.github.com/users/octocat/orgs",
  "repos_url": "https://api.github.com/users/octocat/repos",
  "events_url": "https://api.github.com/users/octocat/events{/privacy}",
  "received_events_url": "https://api.github.com/users/octocat/received_events",
  "type": "User",
  "site_admin": false,
  "name": "The Octocat",
  "company": "GitHub",
  "blog": "http://www.github.com/blog",
  "location": "San Francisco",
  "email": null,
  "hireable": null,
  "bio": null,
  "public_repos": 7,
  "public_gists": 8,
  "followers": 2070,
  "following": 5,
  "created_at": "2011-01-25T18:44:36Z",
  "updated_at": "2018-01-01T12:31:09Z"
}
```

Appendix C
Data from the GitHub Repo API


```
[
 {
  "id": 18221276,
  "name": "git-consortium",
  "full_name": "octocat/git-consortium",
  "owner": {
   "login": "octocat",
   "id": 583231,
   "avatar_url": "https://avatars3.githubusercontent.com/u/583231?v=4",
   "gravatar_id": "",
   "url": "https://api.github.com/users/octocat",
   "html_url": "https://github.com/octocat",
   "followers_url": "https://api.github.com/users/octocat/followers",
   "following_url": "https://api.github.com/users/octocat/following{/other_user}",
   "gists_url": "https://api.github.com/users/octocat/gists{/gist_id}",
   "starred_url": "https://api.github.com/users/octocat/starred{/owner}{/repo}",
   "subscriptions_url": "https://api.github.com/users/octocat/subscriptions",
   "organizations_url": "https://api.github.com/users/octocat/orgs",
   "repos_url": "https://api.github.com/users/octocat/repos",
   "events_url": "https://api.github.com/users/octocat/events{/privacy}",
   "received_events_url": "https://api.github.com/users/octocat/received_events",
   "type": "User",
   "site_admin": false
  },
  "private": false,
  "html_url": "https://github.com/octocat/git-consortium",
  "description": "This repo is for demonstration purposes only.",
  "fork": false,
  "url": "https://api.github.com/repos/octocat/git-consortium",
  "forks_url": "https://api.github.com/repos/octocat/git-consortium/forks",
  "keys_url": "https://api.github.com/repos/octocat/git-consortium/keys{/key_id}",
  "collaborators_url": "https://api.github.com/repos/octocat/git-consortium/collaborators{/collaborator}",
  "teams_url": "https://api.github.com/repos/octocat/git-consortium/teams",
  "hooks_url": "https://api.github.com/repos/octocat/git-consortium/hooks",
  "issue_events_url": "https://api.github.com/repos/octocat/git-consortium/issues/events{/number}",
  "events_url": "https://api.github.com/repos/octocat/git-consortium/events",
  "assignees_url": "https://api.github.com/repos/octocat/git-consortium/assignees{/user}",
  "branches_url": "https://api.github.com/repos/octocat/git-consortium/branches{/branch}",
  "tags_url": "https://api.github.com/repos/octocat/git-consortium/tags",
  "blobs_url": "https://api.github.com/repos/octocat/git-consortium/git/blobs{/sha}",
  "git_tags_url": "https://api.github.com/repos/octocat/git-consortium/git/tags{/sha}",
  "git_refs_url": "https://api.github.com/repos/octocat/git-consortium/git/refs{/sha}",
  "trees_url": "https://api.github.com/repos/octocat/git-consortium/git/trees{/sha}",
  "statuses_url": "https://api.github.com/repos/octocat/git-consortium/statuses/{sha}",
  "languages_url": "https://api.github.com/repos/octocat/git-consortium/languages",
```

```
  "stargazers_url": "https://api.github.com/repos/octocat/git-consortium/stargazers",
  "contributors_url": "https://api.github.com/repos/octocat/git-consortium/contributors",
  "subscribers_url": "https://api.github.com/repos/octocat/git-consortium/subscribers",
  "subscription_url": "https://api.github.com/repos/octocat/git-consortium/subscription",
  "commits_url": "https://api.github.com/repos/octocat/git-consortium/commits{/sha}",
  "git_commits_url": "https://api.github.com/repos/octocat/git-consortium/git/commits{/sha}",
  "comments_url": "https://api.github.com/repos/octocat/git-consortium/comments{/number}",
  "issue_comment_url": "https://api.github.com/repos/octocat/git-
consortium/issues/comments{/number}",
  "contents_url": "https://api.github.com/repos/octocat/git-consortium/contents/{+path}",
  "compare_url": "https://api.github.com/repos/octocat/git-consortium/compare/{base}...{head}",
  "merges_url": "https://api.github.com/repos/octocat/git-consortium/merges",
  "archive_url": "https://api.github.com/repos/octocat/git-consortium/{archive_format}{/ref}",
  "downloads_url": "https://api.github.com/repos/octocat/git-consortium/downloads",
  "issues_url": "https://api.github.com/repos/octocat/git-consortium/issues{/number}",
  "pulls_url": "https://api.github.com/repos/octocat/git-consortium/pulls{/number}",
  "milestones_url": "https://api.github.com/repos/octocat/git-consortium/milestones{/number}",
  "notifications_url": "https://api.github.com/repos/octocat/git-
consortium/notifications{?since,all,participating}",
  "labels_url": "https://api.github.com/repos/octocat/git-consortium/labels{/name}",
  "releases_url": "https://api.github.com/repos/octocat/git-consortium/releases{/id}",
  "deployments_url": "https://api.github.com/repos/octocat/git-consortium/deployments",
  "created_at": "2014-03-28T17:55:38Z",
  "updated_at": "2017-12-06T01:15:32Z",
  "pushed_at": "2016-10-30T13:43:30Z",
  "git_url": "git://github.com/octocat/git-consortium.git",
  "ssh_url": "git@github.com:octocat/git-consortium.git",
  "clone_url": "https://github.com/octocat/git-consortium.git",
  "svn_url": "https://github.com/octocat/git-consortium",
  "homepage": null,
  "size": 190,
  "stargazers_count": 8,
  "watchers_count": 8,
  "language": null,
  "has_issues": true,
  "has_projects": true,
  "has_downloads": true,
  "has_wiki": true,
  "has_pages": false,
  "forks_count": 27,
  "mirror_url": null,
  "archived": false,
  "open_issues_count": 4,
  "license": {
    "key": "mit",
    "name": "MIT License",
    "spdx_id": "MIT",
    "url": "https://api.github.com/licenses/mit"
```

```
  },
  "forks": 27,
  "open_issues": 4,
  "watchers": 8,
  "default_branch": "master"
 },
 ...
]
```