IJoL International Journal of Librarianship

ISSN: 2474-3542 Journal homepage: http://journal.calaijol.org

Develop an Interactive Python Dashboard for Analyzing EZproxy Logs

Andy Huff, Matthew Roth, Weiling Liu

Abstract:

This paper describes the development of an interactive dashboard in Python with EZproxy log data. Hopefully, this dashboard will help improve the evidence-based decision-making process in electronic resources management and explore the impact of library use.

To cite this article:

Huff, A., Roth, M., & Liu, W. (2024). Develop an Interactive Python Dashboard for Analyzing EZproxy Logs. *International Journal of Librarianship*, *9*(1), 59-73. https://doi.org/10.23974/ijol.2024.vol9.1.374

INTERNATIONAL JOURNAL OF LIBRARIANSHIP, 9(1), 59-73
ISSN: 2474-3542

Develop an Interactive Python Dashboard for Analyzing EZproxy Logs

Andy Huff, Matthew Roth, Weiling Liu
University of Louisville, Louisville, Kentucky,United States

ABSTRACT

This paper describes the development of an interactive dashboard in Python with EZproxy log data. Hopefully, this dashboard will help improve the evidence-based decision-making process in electronic resources management and explore the impact of library use.

Keywords: Academic libraries, Interactive dashboard, Python, Collection management, Information resources management, Data visualization, EZproxy server logs

INTRODUCTION

The University of Louisville is an urban research university located approximately three miles from downtown Louisville. The University of Louisville Library system consists of the following libraries: Margaret Bridwell Art Library, William F. Ekstrom Library, Kornhauser Health Sciences Library, Louis D. Brandeis School of Law Library, Dwight Anderson Music Library, and Archives and Special Collections (ASC). Only Art, Ekstrom, Kornhauser, Law, and Music have online resource subscriptions. Ekstrom Library manages most electronic resources for all libraries and subject-related ones for the Art Library. The other libraries manage their subject-related resource subscriptions. Access to all the subscribed resources is restricted by IP address ranges. A locally hosted proxy server, EZproxy, is used for remote access to all the online resources.

We use Springshare's LibInsight to centrally collect and manage all library usage data, from gate counts to electronic resource usage. However, the e-resources usage data received from the vendors track metrics such as the number of searches or downloads, and cost of use, etc. Such statistical reports are good for decision-making on whether to keep or drop a subscription. It is obvious that collecting search statistics from vendors is not enough anymore. The library administrators in academic libraries are often asked questions like what impact a library has on students' success. What library usage data may help in telling this kind of story? In answering these kinds of questions, Montenegro et al. 2016 conducted a study on the relationship between the use of library e-resources, access to library physical resources, and students' academic performance. They used three datasets, one from the library information system about library loans, one from the university system on students' academic performance, and one from EZproxy about the use of online electronic resources. The results of the study show that access to electronic resources has a greater impact on performance than the number of library items borrowed. There is another study with EZproxy usage data: Kabo et al. 2023 on the longitudinal associations between the use of

library electronic resources and students' performance. We have EZproxy logs which include some helpful information but have not been used for anything. The authors intended to develop a tool for analyzing the EZproxy data to see if they can show any trends within that data.

LITERATURE REVIEW

As some articles reveal, EZproxy logs with enhanced data can be used to analyze the usage of e-resources to answer questions like who the users are: students, graduate students, or faculty; whether the users are on campus or off campus; what kind of devices the users are using for the access; or how the users find the resources (Dennison and Sung, 2019; Joseph, et al., 2019; Kohler and Stoval, 2019; Yeager, 2017).

However, "[I]t can often be difficult to analyze EZproxy data using the conventional tools available (Gonzales, 2018)." Gonzales (2018) pointed out in the article about EZproxy log analysis that "[u]sing the Python programming language with its collection of modules created specifically for data analysis can help with this task, and ultimately result in better and more useful data customized to the needs of the library using it." Davis 2014 also described how Python was used in analyzing EZproxy logs in the article, "Analyzing EZproxy logs with Python."

As to how the analysis results are presented, using an interactive visualization dashboard seems to demonstrate its advantages. Archambault et al. 2015, p. 1 argue that a framework for meaningful data visualization has merit:

> There are advantages to presenting data visually rather than as a set of flat statistics. Proper data visualization facilitates the recognition of patterns and relationships to communicate a message in a more compelling and interesting way. It allows the complexity of the data to be understood more easily.

Murphy 2019 also concluded in the article about analyzing EZproxy logs with Tableau, a visualization tool, that the real-time data provided by this process would enable librarians and library staff to quickly derive actionable insights from EZproxy data. Such data will enable librarians and library staff to share the library's positive impact on researcher's lives. As Chen et al. 2022 found out, library data presented in dashboards could play a crucial role in assisting in the improvement of library collection utilization and collection development besides providing insights.

METHODOLOGY

Our goal was to build an interactive dashboard. A dashboard is a type of graphical user interface which provides users with an at-a-glance view of relevant information. An interactive dashboard adds a more visual element to presentations. This means users can more easily understand the numbers, ramifications, and outcomes of the data. Although we did not use any of their specific codebases, we were inspired by Dunder Data COVID-19 Dashboard project (https://www.dunderdata.com/blog/build-an-interactive-data-analytics-dashboard-with-python-a-comprehensive-course). We used some of the same Python packages and building blocks to build

a prototype interactive dashboard with our dataset. We developed the dashboard in Python and Jupyter Notebooks with one month's EZproxy log data. We keep daily logs of all EZproxy access and store them on our EZproxy server which are accessible by authorized personnel only as needed. The sample log data was transformed using ezPAARSE and imported into the Pandas library for cleaning and reformatting before use. The final dashboard was presented on a local web server using the Dash Python library and Plotly for data visualization. During the development of this prototype, we also used the same dataset in LibInsight and Tableau Public for a brief comparison test. This study was approved by the University of Louisville's Institutional Review Board (IRB number: 23.0687).

SYSTEM SETTINGS

We ran the analyses in a virtual instance of Ubuntu 22.04 LTS using the default window manager, GNOME 42.5. The server was hosted on a local vSphere virtual infrastructure provisioned with 500GB of hard disk space. The software used was Python 3.10.6, Docker Desktop, LibreOffice Calc 7.3.7.2 (and Microsoft Excel using Microsoft 365 on another machine). We also used ezPAARSE version 3.7.4, running within a container on Docker Desktop. We also ran analyses on a Windows 11 version 22H2 laptop with an 11th gen core i7 processor, 32GB of RAM, and a 512GB SSD. The laptop was used to create our Tableau dashboard using Tableau Public 2022.4.

DATA PREPARATION

We used a one-month EZproxy log from January 2019 as the sample for analysis. The raw EZproxy data header includes %h %l %u %t "%r" %s %b %{referer}i %{EZproxy-session}i. The interpretation of the header is as follows:

Log Header	Description	Notes
%h	IP address of host accessing EZProxy	
%l	Remote username obtained by identid (if identid is not used a - will be inserted)	It is blank in our dataset
%u	Username used to log into EZproxy	
%t	Date/time of request	in format 01/Jan/2023:21:09:58 –0400
%r	Complete request	e.g. GET http://www.somedb.com HTTP/1.0
%s	HTTP numeric status of request	e.g. 302, 200
%b	number of bytes transferred	e.g. 312399
%{referer}i	The URL of the website the user was on prior to accessing EZproxy.	
%{ezproxy-session}i	EZproxy identifier for the user's current session	e.g. voSShqdaBIJlH

Figure 1. EZpoxy log header

Our log file gives us a complete representation of who a user is and what a user was accessing, at what time they gained access, whether they were successful in accessing the material requested, and the size of the file or page that was accessed. Figure 2 shows an obfuscated screenshot of the original log before cleaning. Now we need to clean up the data and turn it into something a little more useful.

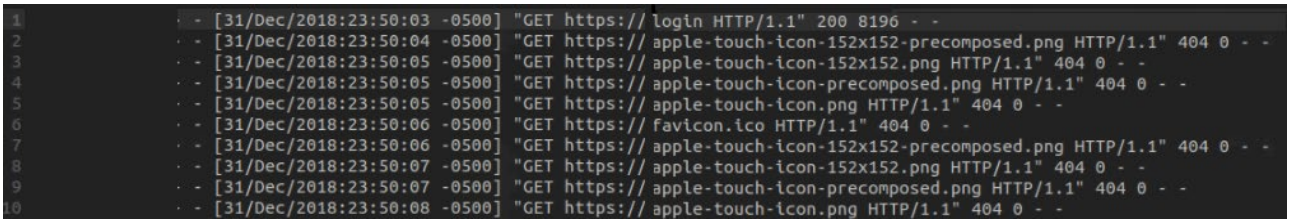

Figure 2. EZproxy data log in its raw form without sensitive information

DATA CLEANING

To clean the data, we first ran the raw log through ezPAARSE, an open-source software program that uses an API to parse data. ezPAARSE is a powerful free, open-source software developed by an international community of users that can be used to parse EZproxy logs as well as Apache and Squid logs. ezPAARSE allows you to specify and design your log file type and upload your raw EZproxy logs to a locally run docker-compose instance of ezPAARSE. There is also an official demo of ezPAARSE at demo.ezpaarse.org. ezPAARSE helped us not only clean the data but also

enhanced the process of identifying databases and platforms accessed. Through this process, we removed redundant and unnecessary entries, for example, downloads of website artifacts such as website favicons and JavaScript. This left us with entries of database searches, HTML article accessions, and PDF article accessions. With ezPAARSE we also identified more detailed database information, including ISSN and DOI information, publication dates, subject. To obtain information about user types, we used Microsoft Excel to match user IDs within that data to another dataset of user information from our integrated library system (ILS). These user IDs were then dropped from the final dataset, leaving us with just their status. Some examples of status are faculty, law faculty, staff, graduate student, undergraduate student, and retirees.

After the cleaning and matching, we removed the fields that may reveal personal information as well as time of access, such as: datetime (showed specific time of access down to the second), Login (%u), Platform, Log_id (%l), EZproxy_session (%{EZproxy-session}I), Status (%s), Size (%b), host, identd and Referer (%{referer}i). We also removed DOI and information that could uniquely identify an article that was accessed. Platform and Referer columns were duplicated after the ezPAARSE process was completed and were deemed unnecessary for this example dataset. While EZproxy_session would be helpful to see how many resources were used by an individual during a single session, for the purpose of this test we decided to exclude it; instead opting to leave just their status within the university as the only potential link to the user. This results in a dataset we can run through the Plotly python library for creating charts, using those in DASH and creating our dashboard. Figure 3 shows the examples of cleaned data.

	A	B	C	D	E	F	G	H	I	J
1	date	status	platform_name	publisl	rtype	mime	print_i	online	publication_title	publication_d
2	1/1/2019	Faculty	Wiley	Wiley	ABS	HTML	1091-426	1520-639	Depression and Anxiety	2016
3	1/1/2019	Faculty	Wiley	Wiley	ARTICLE	HTML	1091-426	1520-639	Depression and Anxiety	2016
4	1/1/2019	Undergrac	Karger	S. Karger	ARTICLE	HTML	0033-319	1423-034	Psychotherapy and Psychosomatics	2013
5	1/1/2019	Staff	Science Direct	Elsevier	ARTICLE	HTML	0898-6568		Cellular Signalling	2019
6	1/1/2019	Graduate	Science Direct	Elsevier	ARTICLE	HTML	1535-610	1878-368	Cancer Cell	2018
7	1/1/2019	Faculty	Gale Cengage		ARTICLE	HTML				
8	1/1/2019	Staff	Gale Cengage		ARTICLE	HTML				
9	1/1/2019	Staff	Wiley	Wiley	ABS	HTML	1742-783	1742-784	Basic & Clinical Pharmacology & Toxic	2009
10	1/1/2019	Staff	Wiley	Wiley	ARTICLE	HTML	1742-783	1742-784	Basic & Clinical Pharmacology & Toxic	2009
11	1/1/2019	Staff	Wiley	Wiley	ABS	HTML	0279-075	1468-011	Pacific Philosophical Quarterly	1997
12	1/1/2019	Staff	Wiley	Wiley	TOC	MISC	1523-089	1708-820	Clinical Implant Dentistry And Related Research	null
13	1/1/2019	Staff	Wiley	Wiley	TOC	MISC	1523-089	1708-820	Clinical Implant Dentistry and Related Research	
14	1/1/2019	Faculty	Wiley	Wiley	TOC	MISC	0303-697	1600-051	Journal Of Clinical Periodontology	null
15	1/1/2019	Faculty	Wiley	Wiley	TOC	MISC	1523-089	1708-820	Clinical Implant Dentistry And Related Research	null
16	1/1/2019	Undergrac	Wiley	Wiley	TOC	MISC	1523-089	1708-820	Clinical Implant Dentistry and Related Research	
17	1/1/2019	Faculty	Web of Science		SEARCH	HTML				
18	1/1/2019	Faculty	UpToDate		SEARCH	HTML				
19	1/1/2019	Faculty	UpToDate		SEARCH	HTML				
20	1/1/2019	Staff	Wiley	Wiley	ABS	HTML	0905-716	1600-050	Clinical Oral Implants Research	2018
21	1/1/2019	Faculty	Wiley	Wiley	ARTICLE	HTML	0905-716	1600-050	Clinical Oral Implants Research	2018
22	1/1/2019	Faculty	Wiley	Wiley	ARTICLE	PDF	0905-716	1600-050	Clinical Oral Implants Research	2018
23	1/1/2019	Staff	Wiley	Wiley	ABS	HTML	0022-349	1943-367	Journal of Periodontology	2018
24	1/1/2019	Staff	Wiley	Wiley	ARTICLE	HTML	0022-349	1943-367	Journal of Periodontology	2018
25	1/1/2019	Faculty	Wiley	Wiley	ARTICLE	PDF	0022-349	1943-367	Journal of Periodontology	2018
26	1/1/2019	Faculty	Science Direct	Elsevier	ARTICLE	HTML	0272-6386		American Journal of Kidney Diseases	2019
27	1/1/2019	Faculty	Wiley	Wiley	TOC	MISC	0022-349	1943-367	Journal of Periodontology	
28	1/1/2019	Faculty	Wiley	Wiley	TOC	MISC	0022-349	1943-367	Journal of Periodontology	
29	1/1/2019	Faculty	Wiley	Wiley	ARTICLE	PDF	1600-613	1600-614	American Journal of Transplantation	2019
30	1/1/2019	Staff	Wiley	Wiley	ARTICLE	PDF	0022-349	1943-367	Journal of Periodontology	2018
31	1/1/2019	Staff	Wiley	Wiley	ARTICLE	HTML	0022-349	1943-367	Journal of Periodontology	2018
32	1/1/2019	Faculty	Wiley	Wiley	ARTICLE	PDF	1600-613	1600-614	American Journal of Transplantation	2019
33	1/1/2019	Staff	Wiley	Wiley	TOC	MISC	0022-349	1943-367	Journal of Periodontology	
34	1/1/2019	Staff	Wiley	Wiley	ARTICLE	HTML	0022-349	1943-367	Journal of Periodontology	2018
35	1/1/2019	Undergrac	Wiley	Wiley	ARTICLE	PDF	1600-613	1600-614	American Journal of Transplantation	2019
36	1/1/2019	Staff	Wiley	Wiley	ARTICLE	PDF	1600-613	1600-614	American Journal of Transplantation	2019
37	1/1/2019	Staff	Wiley	Wiley	ARTICLE	PDF	1600-613	1600-614	American Journal of Transplantation	2019

Figure 3. Cleaned Data

DEVELOPMENT

The tool used for this development is Python with Jupyter Notebook, Pandas, Plotly, and DASH. Jupyter Notebook allows us to easily comment and edit our code while keeping it readable for other programmers. It creates code blocks which can be toggled between executable code and code commentary. It also allows us to save it as a self-contained file with the caveat that it can be read only by the Jupyter Notebook application. Pandas is a data science Python library that allows us to manipulate datasets, which pandas call data frames. Pandas is built on NumPy, which is a python package that handles numerical operations such as linear algebra on large datasets; specifically, arrays. Pandas can import data from various sources and with multiple different delimiters. Pandas can also read the Excel file thanks to a function called read excel. This allows Pandas to ingest the Excel file with just the filename and a specified separator (i.e. ";"). Further, Pandas allows you to manipulate your data, rearrange columns and save your modified datasets as CSV, JSON, SQL, Excel files and more. Plotly is a Python library that allows us to create visualizations with relative ease, provided we know the syntax. Plotly also allows us to make beautiful interactive visualizations without knowing much code. Within a Ploty visualization you can zoom in and out on data, pan your graph, select areas of your graph to zoom into and choose which data points you want visible on the graph, and it will dynamically update based on your input. DASH is a Python library that allows us to create responsive websites using Python code. It can also take HTML and CSS code and parse it into a website. We used a Python virtual environment (venv) to manage our packages and keep our project self-contained. We created the venv using the following code:

```
python –m  venv ~./EZproxy_venv  [For Linux]
python –m venv EZProxy_venv [For Windows]
```

While we have created a new virtual environment for Python, it does not assume that we want to use it as soon as we create it. So, we need to activate the newly created EZproxy_venv python instance so that we can run our python code. To do so, we need to run the activate command within the bin folder of the EZproxy_venv folder that was created when we created the new virtual environment. To run the activate command, we ran the following code:

```
source ~./EZproxy_venv/bin/activate [For Linux]
EZProxy_venv\Scripts\activate.bat [For Windows]
```

Now in our terminal we see that we are in our python environment when we see (EZproxy_venv) before our user and machine information (see Figure 4.) We used pip to install Jupyter Notebook, Pandas, and Plotly. Once they were installed, we ran Juptyer Notebook by issuing the following command:

```
jupyter notebook
```

Figure 4 shows the screenshot of the code execution status. After the environment was set up, we created a new notebook and imported our Python libraries into it (see Figure 5.) From there, we used Pandas to import our dataset.

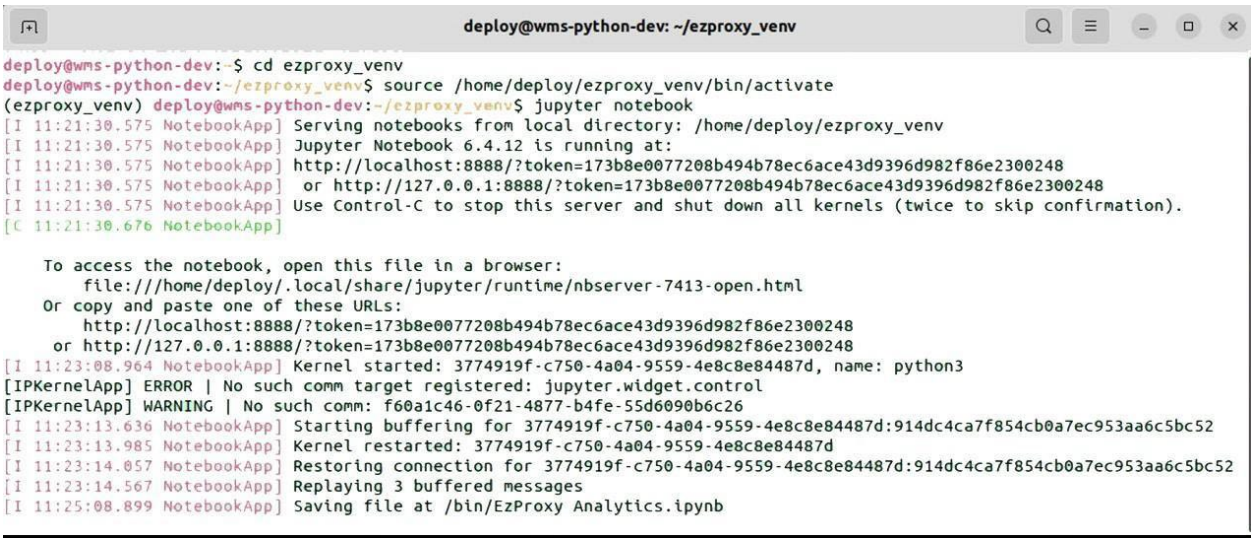

Figure 4. Activating Python and the Jupyter Notebook

```
In [1]: #Importing the libraries we are going to use for our analysis
        import pandas as pd
        import numpy as np

        import plotly.express as px
        import plotly.graph_objects as go

        import dash
        from dash import html
        from dash import dcc

        ezproxy_df = pd.read_excel('EZProxy_Data_with_status.xlsx')
```

Figure 5. Sample code of the Jupyter Notebook detailing how the data is imported

Once imported, we inspected our dataset to ensure it was correctly ingested using the head() function.

To generate a report on the database usage by patron types, we selected the fields, Status (patron types), Platform_name (databases) and Datetime. Figure 6 illustrates the codes and outcome of this report in a bar graph. Code that has a pound sign in front of the line of code is commented out and not run. To easily make a website with our dataset, we used the DASH package with Python 3.5.8 as well as the Plotly, Pandas, and NumPy libraries for data analysis and manipulation. We used Plotly to make a few simple example data visualizations. The visualizations are straightforward and easy to create (Figure 6 and 7).

```
In [3]: ezproxy_status = ezproxy_df[["status","platform_name","datetime"]]
        ezproxy_status = ezproxy_status.groupby(["status","platform_name"])['datetime'].count().reset_index()
        ezproxy_status.columns=["status", "platform_name","count"]
        #ezproxy_status = ezproxy_status.sort_values(by = ["status","count"], ascending = [True,False])
        #ezproxy_top_100 = ezproxy_status.head(100)
        #print(ezproxy_top_100)

In [4]: chart0 = px.bar(ezproxy_status, x='platform_name', y='count', color="status", title="Accessions by Database and Statu
        chart0.show()
```

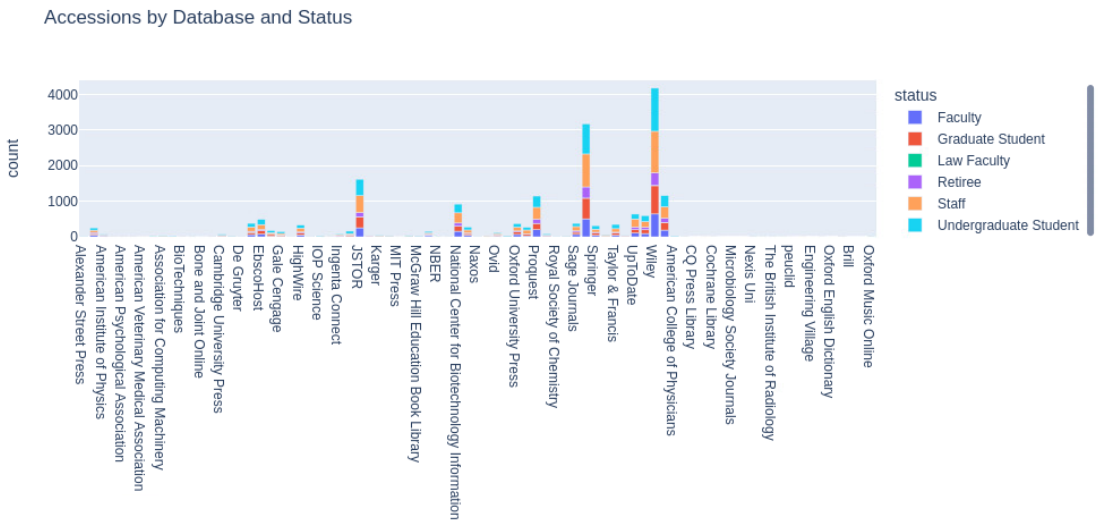

Figure 6. Stacked histogram of database accessions by status

VISUALIZATION

Since the chart shown in Figure 6 is a static graph, what you can see is what is displayed. However, with an interactive dashboard, you can see the additional databases that were not able to be displayed in the current view. By panning, you can scroll across the chart and see all the data. In addition, the status legend is clickable and when you click on one of the entries, the chart is updated to exclude that data and it is grayed out in the legend (see Figure 7).

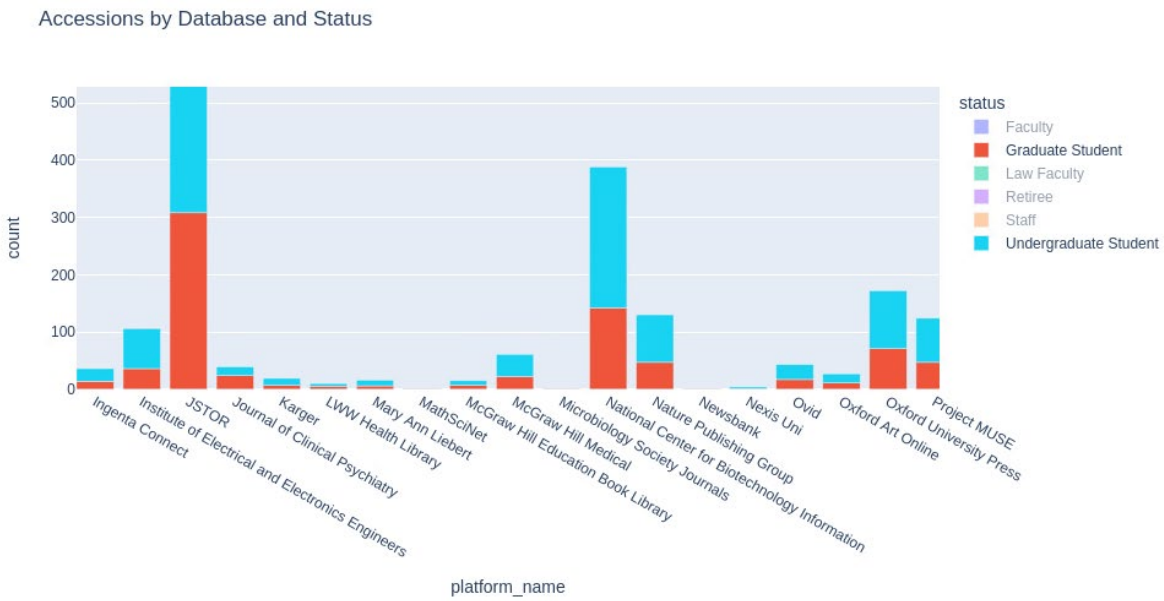

Figure 7. Selecting specific users and zooming in on the data

You can also hover over a data point in your dataset to see the exact count of that specific status (see Figure 8).

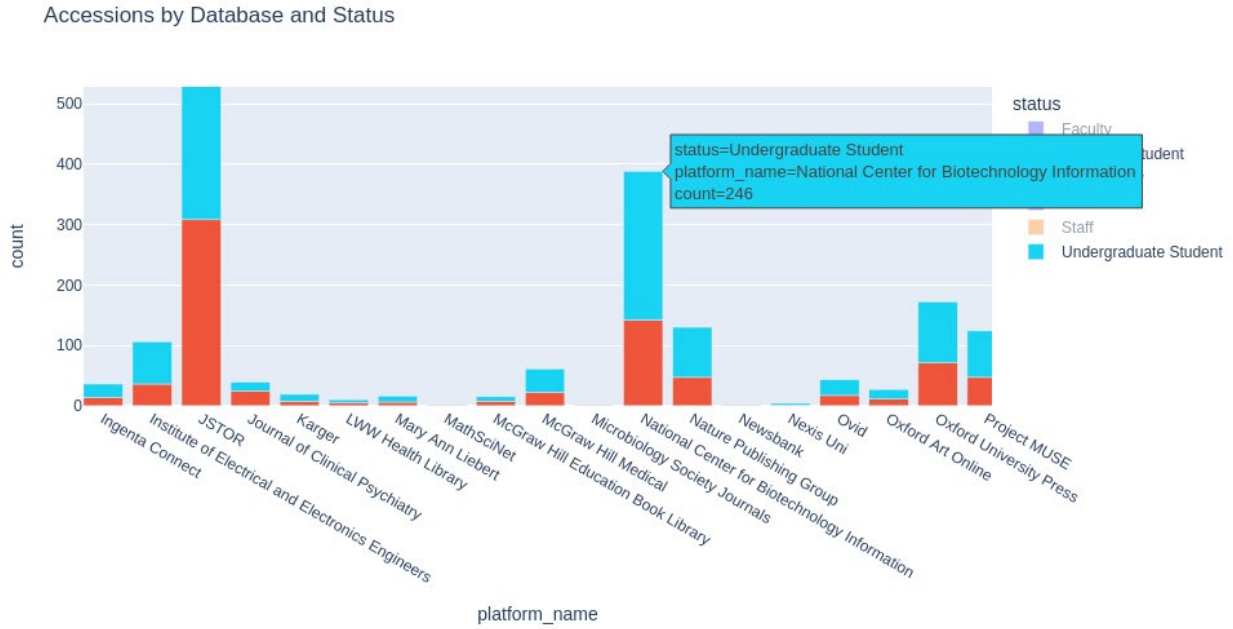

Figure 8. A tooltip that appears when hovering over a bar in the histogram

With Plotly you can also save the current area of the visualization shown as a PNG image (see Figure 9) while you still could quickly go back to a default version of the visualization so that you can examine another aspect of it more easily.

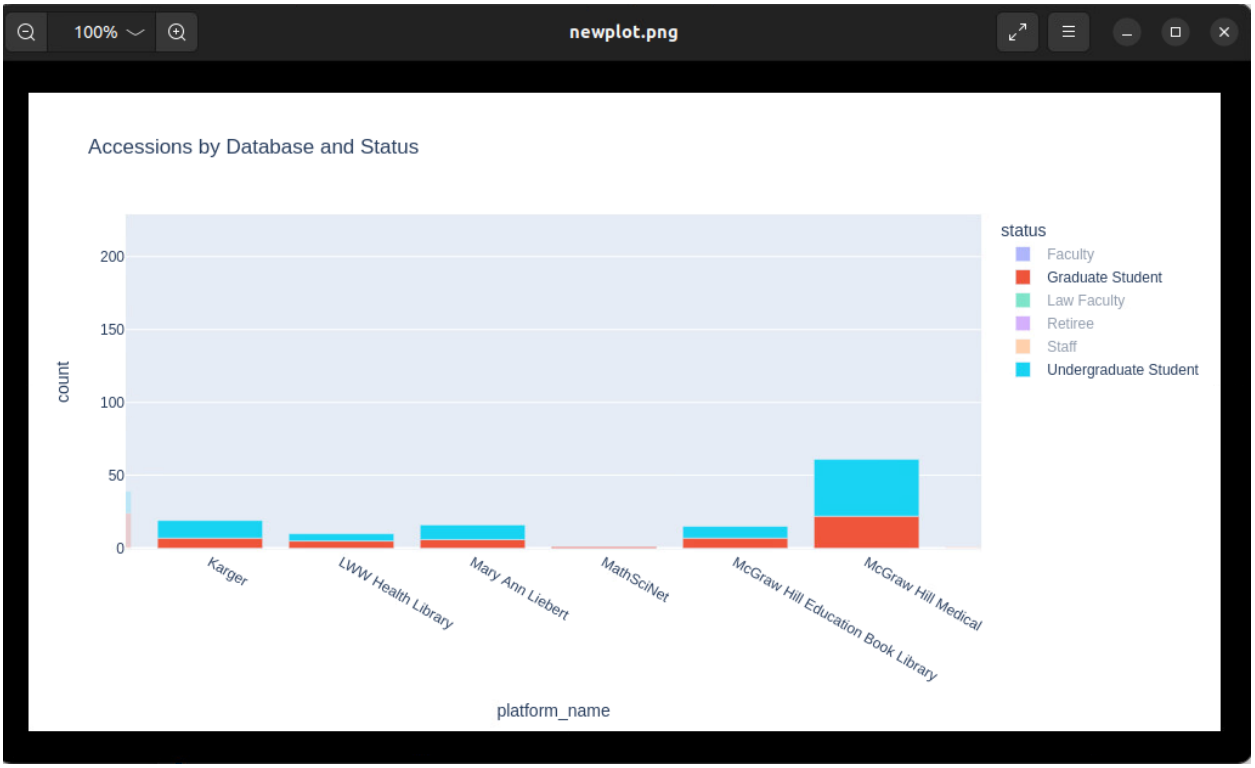

Figure 9. A saved plot from Plotly

Plotly is such a powerful tool for data visualization and exploration. Using Plotly will help us easily see trends in our data when applied to a larger dataset.

Our DASH website is minimal and only includes two visualizations in this prototype as enough to demonstrate the possibilities of the system. Further time and development would need to be invested in the project to make it fully featured and useful to library faculty. Our dashboard only has two visualizations, without any additional CSS or HTML code. These visualizations can be manipulated by a user and can be zoomed into, filtered, and reset just as when we were coding the visualizations themselves in our jupyter notebook.

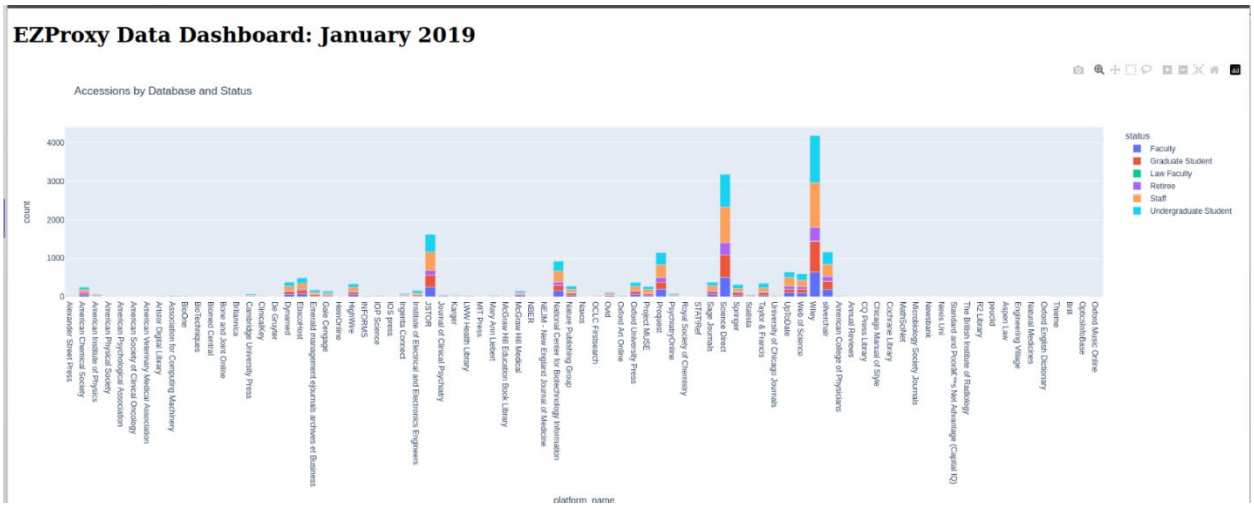

Figure 10. Sample dashboard written in DASH with example histogram.

COMPARISON

Using DASH allowed us to create interactive visualizations with Python. We wondered how we could get similar results using the tools already being used in our Libraries. We chose to compare our DASH website with the capabilities of LibInsight and Tableau Public. Since we already have LibInsight as a data repository with many qualitative datasets that our libraries have captured, we would like to see if it could help us achieve our analytics goal. We manipulated our dataset and tried to create a dashboard. However, our test made us decide not to use it. LibInsight has some limitations on the data to be uploaded into the repository. First, files over 18MB in size are not allowed. In addition, it has specific data types that it requires our data to adhere towards to be accepted. LibInsight must also have a specific valid date and time stamp. Our current dataset doesn't have that specific format by default, and it would need to be further manipulated to get the correct date and time format for the required field. The end result of our test was that most of our data was changed to a text type for acceptable ingestion. In doing so, however, any attempts at dashboard creation are fruitless. The system behaves as if there are no records available for us to parse, and all charts created are empty. As a result, we felt that LibInsight isn't the best solution for our specific use case because it would require additional manipulation of our data to get to the same level that Python has already reached for us.

Tableau is a visual analytics platform, Tableau Desktop and Tableau Public are the major products of Tableau. Tableau Desktop is a fee-based version, and Tableau Public is the simpler free version for users to explore, create, and publicly share data visualizations online.

Using Tableau Public

In our test, we imported our dataset into Tableau Public. Tableau gave us a lot of flexibility in how we could import our data and has abilities to create relational links between datasets before

analysis. We did not save our data while creating this dashboard, but we did make a close approximation to what our Python dashboard was in Tableau. Tableau made it easy for us to do this because it also suggested visualizations for our data based on the fields that we chose to display.

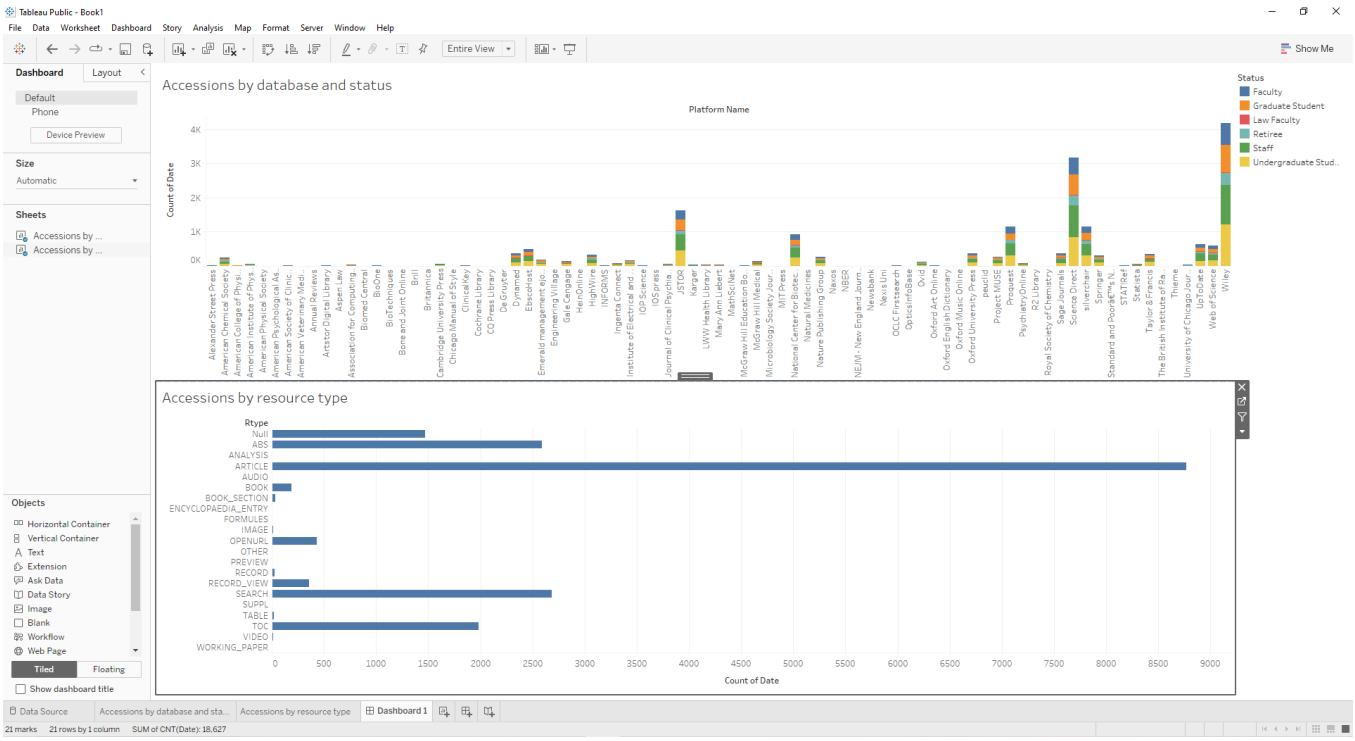

Figure 12. Sample dashboard using Tableau Public

As you can see in Figures 4 and 5, while the data that they show is the same, the key distinction is the short amount of time, ease of use, and cleanliness of the dashboard when made in Tableau as compared to Python. When using Python, a high degree of skill was required, as well as additional time. The Python code and environment setup took around 2.5 to 3 hours. In Tableau, it took under 30 minutes.

Our first impression of Tableau Public was that Tableau Public is intuitive for an analyst to use and requires no coding ability. Tableau is much more intuitive in allowing the quick creation of visualizations. Further, we can create new visualizations quickly, unlike that we must rerun Python code to get a result in our Python project. The major offset is that you cannot save to Tableau Public without making your dashboard publicly accessible. Our Python dashboard only has the cost of initial setup (about 3 hours of our time), and we retain all ownership and accessibility of our data.

SUMMARY

As the publications we found in our literature review reveal and our prototype interactive dashboard development illustrated, we will be able to use such a platform to enhance the use of library usage data, especially EZproxy logs. Both Python and Tableau are excellent tools for building interactive dashboards, with the caveat that Python requires initial setup before it will be useful. However, for libraries with resources for development, hopefully, our prototype can provide some useful information as a start.

Long term, we see the need to create an encrypted local data repository to store our cleaned EZproxy data. Our plan is to create an encrypted MariaDB server with a database specifically for our data and then query that data for analysis. We will also need to devote time to processing all our EZproxy data. It takes about an hour for ezPAARSE to process a year's worth of logs. We would then need to add the status and remove extraneous columns before uploading it to our repository which would take an as yet unknown amount of time.

Python is a powerful language and can be used for data analysis, but requires a certain set of skills, drive and time to accomplish the same thing that Tableau is able to accomplish. For those who have the time and where budget does not allow, Python is a viable method of analyzing your EZproxy logs. A modified Jupyter Notebook with a fake test set will be published to GitHub at https://github.com/athuff02/ezproxy-dashboard for others to use and adapt in their own projects.

One of the biggest limitations that we face with our data collection is that we are unable to capture statistics for those who use our subscribed databases on campus with this solution, as we do not require users to login to access our databases while they are on campus. We want to continue to work on our dashboard, incorporating features based on faculty feedback on the data currently presented. In addition, we are also going to combine our EZproxy data with data from our CardSmart (EAB Rapid Insight) data repository to better identify the departments and majors that utilize our subscribed databases. By doing this, we hope to do more than just descriptive statistics with our data by doing some exploratory linear regressions on our data to determine the causality between a student's major and database usage. We also need to consider the long-term storage of this data, access and other variables around data retention.

References

Archambault SG, Helouvry J, Strohl B, Williams G. 2015. Data visualization as a communication tool. Library Hi Tech News. 32(2):1–9. doi:10.1108/LHTN-10-2014-0098.

Chen J-A, Tu Y-F, Hwang G-J, Wu J-F. 2022. University librarians' perspectives on an importance-performance analysis of authentication system attributes and their attitudes towards authentication log visualization. The Journal of Academic Librarianship. 48(4):102528. doi:10.1016/j.acalib.2022.102528. [accessed 2023 Feb 16]. https://linkinghub.elsevier.com/retrieve/pii/S0099133322000441.

Davis RC. 2014. Analyzing EZproxy logs with Python – Emerging Tech in Libraries. [accessed 2023 Feb 16]. https://emerging.commons.gc.cuny.edu/2014/04/analyzing-ezproxy-logs-python/.

Dennison C, Sung J. 2019. Finding Hidden Treasures in the Data. In: Proceedings of the 2018 Library Assessment Conference: Building Effective, Sustainable, Practical Assessment: December 5–7, 2018, Houston, TX. Association of Research Libraries. p. 26–39. [accessed 2023 Apr 20]. https://www.libraryassessment.org/wp-content/uploads/2019/09/3-Dennison-Sung-FindingHiddenTreasures.pdf.

Gonzales BM. 2018. Analyzing EZproxy SPU Logs Using Python Data Analysis Tools. The Code4Lib Journal.(42). [accessed 2023 Feb 16]. https://journal.code4lib.org/articles/13918.

Joseph P, Kent AJ, Green PD, Robinson M, Bellenger A. 2019. Analysis of EZproxy server logs to visualise research activity in Curtin's online library. Library Hi Tech. 37(4):845–865. doi:10.1108/LHT-04-2018-0050. [accessed 2023 Feb 16]. https://doi.org/10.1108/LHT-04-2018-0050.

Kabo F, Paulson A, Bradley D, Varnum KJ, Teasley S. 2023 Feb 19. Longitudinal Associations between Online Usage of Library-Licensed Content and Undergraduate Student Performance. doi:10.7302/6979. [accessed 2024 Feb 15]. http://deepblue.lib.umich.edu/handle/2027.42/175845.

Kohler E, Stovall C. 2019. Mining EZProxy Data: User Demographics and Electronic Resources. In: Proceedings of the 2018 Library Assessment Conference: Building Effective, Sustainable, Practical Assessment: December 5–7, 2018, Houston, TX. Association of Research Libraries. p. 728–741. [accessed 2023 Apr 20]. https://www.libraryassessment.org/wp-content/uploads/2019/10/79-Kohler-Stovall-MiningEZProxy.pdf.

Montenegro M, Clasing P, Kelly N, Gonzalez C, Jara M, Alarcón R, Sandoval A, Saurina E. 2016. Library Resources and Students' Learning Outcomes: Do All the Resources Have the Same Impact on Learning? The Journal of Academic Librarianship. 42(5):551–556. doi:10.1016/j.acalib.2016.06.020. [accessed 2023 Feb 16]. https://linkinghub.elsevier.com/retrieve/pii/S0099133316301082.

Murphy SA. 2019. A non-programmers guide to enhancing and making sense of EZ Proxy logs. Performance Measurement and Metrics. 20(3):186–195. doi:10.1108/PMM-08-2019-0034. [accessed 2023 Feb 16]. https://doi.org/10.1108/PMM-08-2019-0034.

Yeager HJ. 2017. Using EZproxy and Google Analytics to Evaluate Electronic Serials Usage. Serials Review. 43(3–4):208–215. doi:10.1080/00987913.2017.1350312. [accessed 2023 Feb 16]. https://www.tandfonline.com/doi/full/10.1080/00987913.2017.1350312.

About the authors

Andrew Huff – Business Intelligence Analyst III of Undergraduate Medical Education (UME) at University of Louisville, Louisville, Kentucky, USA. He provides data analysis and accreditation

support and is currently developing a database for UME using Python and Django. His professional interests include statistical analysis, systems integration, python and Django programming, and data management.

Matthew Roth – IT Suppott Technician II at University of Louisville, Louisville, Kentucky, USA. He provides IT and data analytics support to the Ekstrom, Art, and Music libraries. His professional interests include IT Support, data visualization, and Infrastructure technologies.

Weiling Liu -- Professor and Head of Office of Libraries Technology (OLT) at University of Louisville, Louisville, Kentucky, USA. She manages OLT, provides leadership and coordination on technology projects, including evaluation, planning, and implementation, and supports library systems and applications. Her professional interests include statistical analysis, information retrieval, access, systems integration, e-resources management, resource discovery, project management, and data management. She has published in several peer-reviewed journals, including Kentucky Libraries, OCLC Systems & Services, the Journal of the Medical Library Association, and the International Journal of Librarianship. https://orcid.org/0000-0002-6720-728X