Rate My Professor Dashboard: Visualizing University Instructor Ratings

Connor MacKinnon

1 Motivation

Prospective college students are often overwhelmed by advertisements and colleges striving to attract the maximum amount of applicants. Universities release numerous statistics and promotional claims making it overwhelm for students to compare universities on an object basis. Parsing this information can be difficult for students. Colleges may also attempt to hide their shortcomings from these prospective students.

College students regularly take advantage of the website Rate My Professor to rank the quality of professors and add insights about their teaching styles. This tool then presents ratings to other students to assist them in selecting courses that interest them and fit their needs. They may also filter out classes taught by professors with ratings that they deem unappealing.

By scraping and aggregating this data from Rate My Professor, student opinions can be collected about a college's professors and may be grouped by subject, number of ratings, quality, etc. These unfiltered results may represent the student body's broad opinions on the quality and difficulty of education offered at their institution. Creating an interface to process this data and visualize the opinions surrounding a university may be advantageous for prospective students seeking exclusive information. Additionally, students enrolled at a university may use this tool to consider what major they'd like to choose and understand its perceived quality and difficulty by other students.

2 Methods

Rate My Professor lacks a convenient interface or API for easily retrieving rating information, therefore, the website must be scraped manually to obtain the data. With the library Selenium, the webpage can be visited in a headless browser and parsed through.[4] A page exists that will display the average ratings for each professor at a college, but it requires them to be slowly loaded in batches of 8 at a time. After loading every professor, the site's source code (HTML) can be downloaded for processing. Additionally, all of the URLs and names of universities were scraped to generate a list tracking which universities' instructors haven't been downloaded yet. (Figure 1)

The preprocessing on this data uses a library called Beautiful Soup for scanning through HTML and extracting key information. [3] This quickly transforms the ratings into a dataframe for further processing. The project stores the information in a static Polars dataframe and also a database for querying. This saves time for subsequent users who will only need to populate a database with the existing Polars dataframe files instead of having to visit and slowly scrape webpages every time. The database then organizes all instructors, departments, and schools and allows quick queries to deliver the end user information of interest.

On application startup, a PostgreSQL database is created and populated with the existing college data stored. Storing the dataframe files and running the database is advantageous for

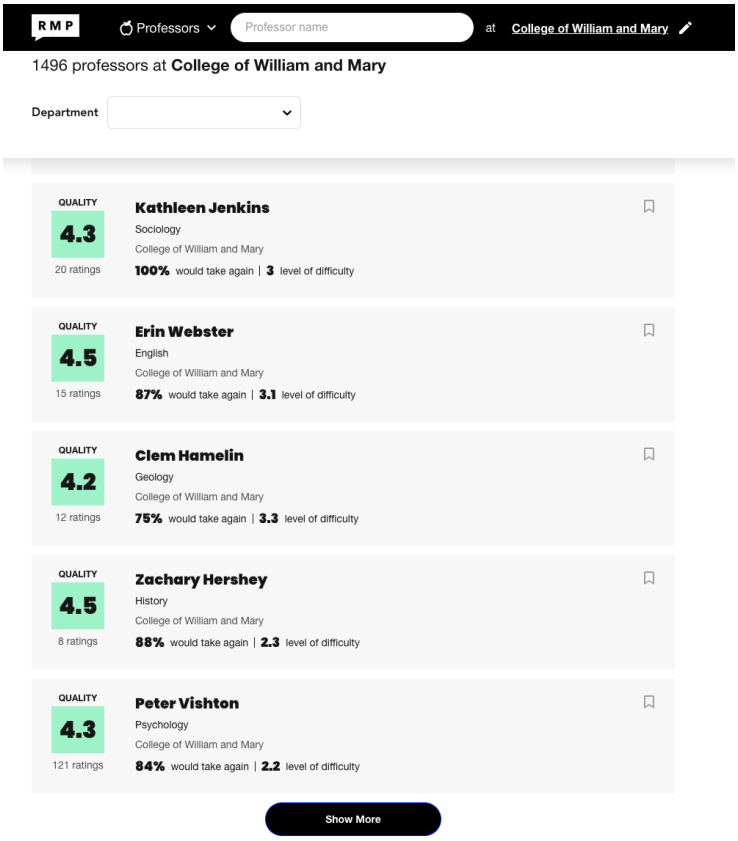

Figure 1: The show more button must be pressed thousands of times

the speed and scaling of the system, along with its portability. The dataframe files are roughly 100 times smaller than storing the HTML files for the webpage of a university. Using a Polars dataframe [5] as opposed to a Pandas dataframe allows processing to be done even faster and for the database to be built faster. Building the database locally is an intentional choice for scaling. Storing a central database file, such as a SQLite database [2], could surpass the file size limit on GitHub and prevent it from being uploaded. Instead, having the dataframe for each schools allows for the database to be quickly built locally. PostgreSQL [1] is used for the database system, given its robustness and scalability in this aspect.

3 Data

The dataset contains the names of schools, their departments, and all the instructors under those departments. Instructors have their names, average quality rating, number of student reviews, the percentage of students who said they would retake that professor, and the average difficulty.(Figure 2) Quality considers the overall deemed effectiveness of a course between one and five (five being the highest quality). The "would take again" percentage looks at whether students indicated they would retake a class or not. This statistic may be a reflection of course content more than the instructor's performance. The average difficulty is the aggregate of scores that students assigned to the professor based on the perceived difficulty between one and five (five being the most challenging). Only the aggregates are being stored as opposed to each raw individual review. Obtaining raw reviews would exponentially slow down the scraping process.

Different institutions have different department names, but any departments with shared

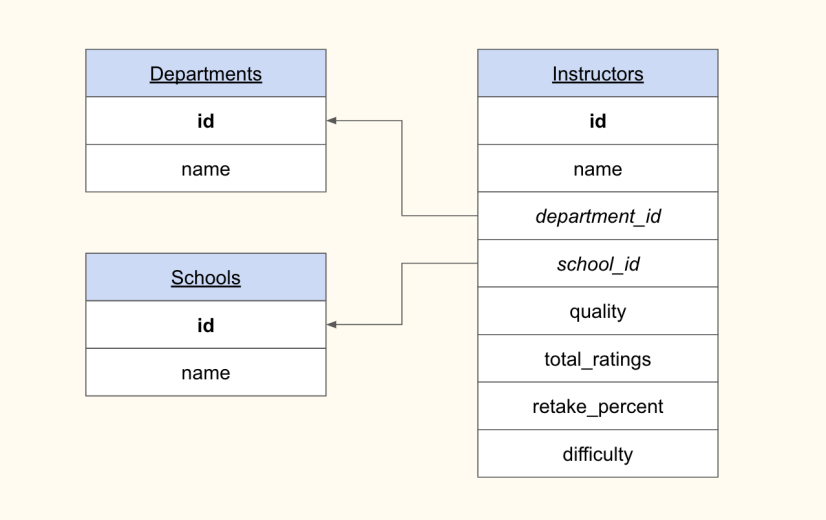

Figure 2: Database Schema

names can be easily compared. Currently, the system does not handle professors working in several departments or at multiple schools in a special manner.

4 Demonstration

This project is an interactive dashboard that visualizes insights about schools. It has the functionality to compare metrics for average difficulties, qualities, and percentage of students who said they would take the class again. These metrics are visualized between departments at a single university and across universities by department.

The individual school metrics allow comparison across departments. This webpage creates Plotly bar plots for all the departments at a university.(Figure 3) These interactive plots allow for zooming and selecting each bar to see more information. The plots may be toggled to show difficulty, quality, or retake percentages. Additionally, plots can set a minimum threshold for department reviews. This allows departments with only a few reviews to be filtered out and not clutter the graph with their skewed representations.

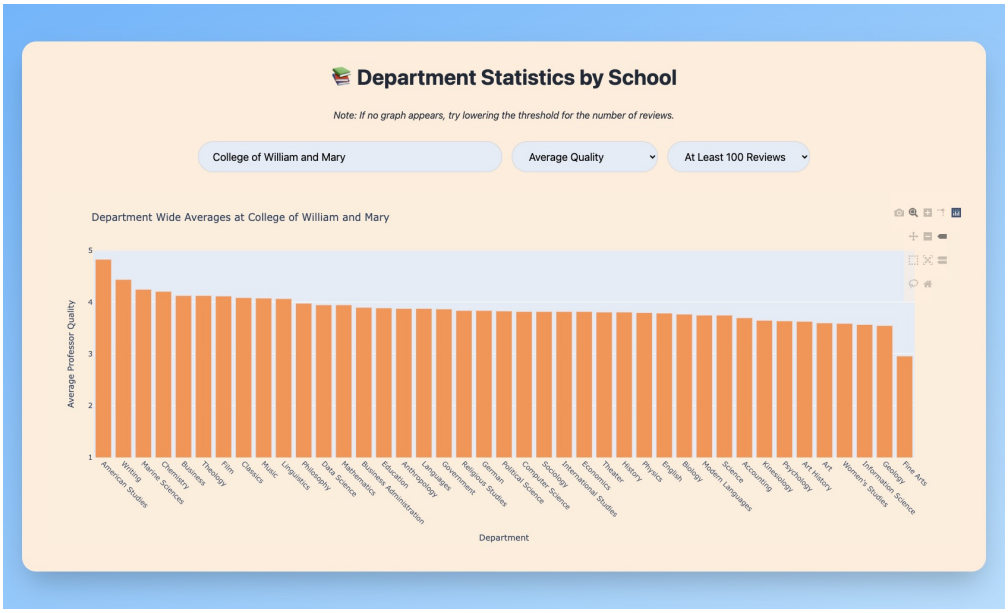

Figure 3: Individual University Departments Page

Comparing across universities is done with several box plots. The user may enter universities they would like to compare and then select the department they want to compare by. They may also choose whether they want to display difficulty, quality, or retake percentages. After making their selection, the web page will display a box plot for the distribution of ratings in that department for each school. Closely grouping these plots allows for comparison of the quantiles and how the medians may differ.(Figure 4) For example, the quality ratings for computer science have a median of 4.1 at Harvard and a 3.9 at William and Mary. Additionally, the selection of majors available depends on which universities are selected. Only departments available at all selected schools will be shown.

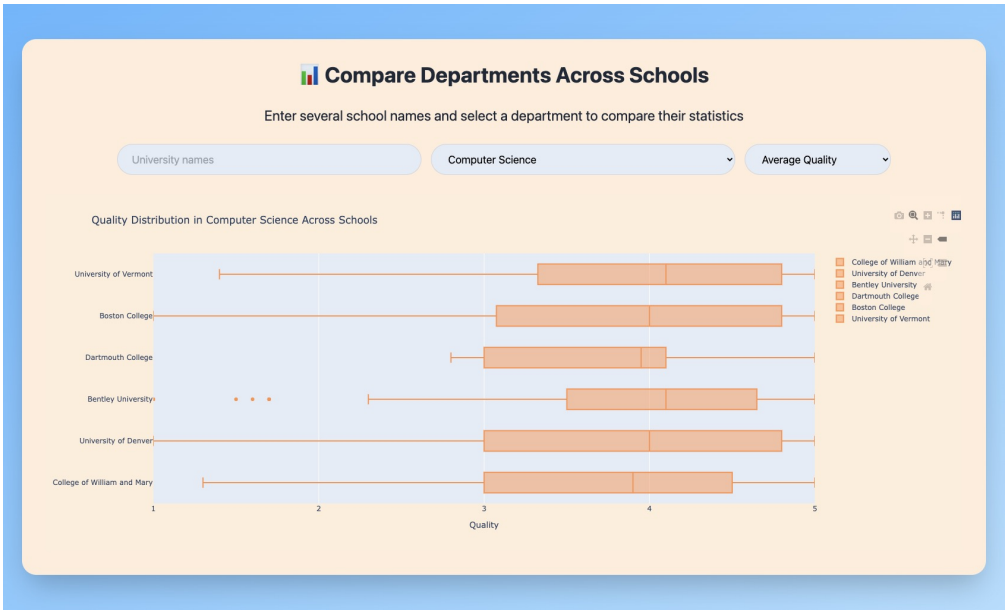

Figure 4: Compare University Departments Page

For schools not loaded in the database, the "Download Additional Universities" page can

be utilized to scrape and add additional data. This page has an autocomplete search bar for schools that have no instructor data downloaded yet. Selecting the download button will start the process and load the school of choice and then extract the instructor ratings and insert everything into the database. Afterwards the data is stored in a lightweight parquet file. Finally, the repository currently contains roughly 350 precomputed schools and is a formidable starting point for comparing colleges and universities. This aims to cover a variety of interesting universities and serve as a starting point for the user to learn and compare. (Figure 5)

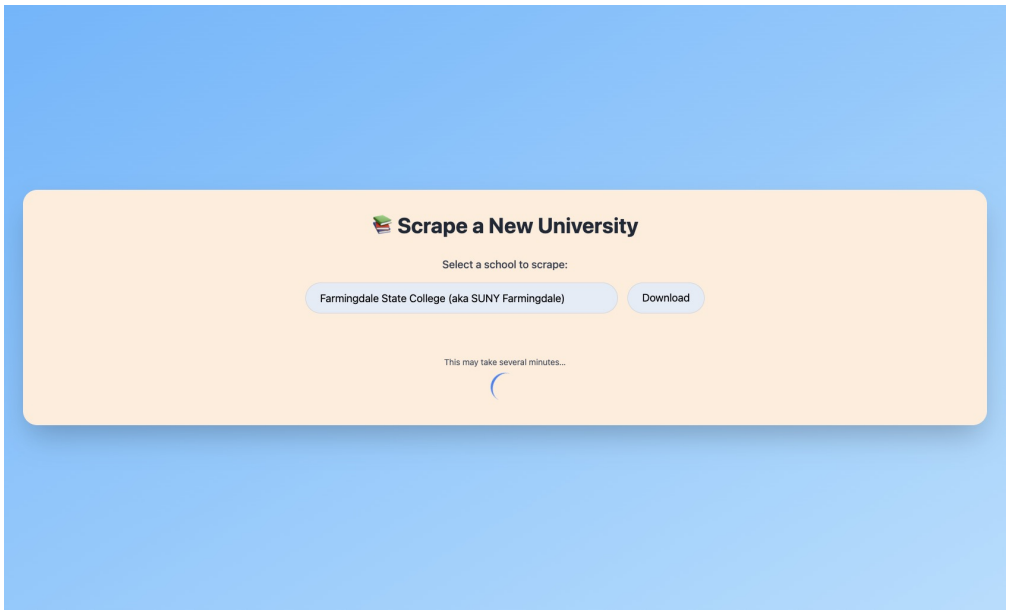

Figure 5: Download Additional Universities Page

5 Future Work

This application has the opportunity to keep expanding in scale and scope. The current trajectory is to add more graphing and available information. Showing individual professors in departments or across a college and highlighting the top performers could help students choose their classes. Scraping more universities would also allow for accurate averages across all universities to develop an understanding what trends exist on macro-levels. For example across all colleges, Creative Writing may have a higher quality if storytelling and writing were perceived as more enjoyable or easier skills to develop. Accurate measurements cannot be taken without a larger sample size of universities though.

Another opportunity for improvement is with updating results. Currently, updating the data for a college has to be done internally by deleting its records and rescraping the college. Implementing a way to refresh a school improve relevancy. Finally, by hosting this database and web-application on a server, the service could be made more readily available and help more students in their academic journeys.

References

[1] PostgreSQL Global Development Group. Postgresql: The world's most advanced open source relational database, 2023.

[2] D. Richard Hipp. Sqlite: Embedded sql database engine, 2023.

[3] Leonard Richardson. Beautiful soup: Html/xml parser for python. https://www.crummy.com/software/BeautifulSoup/, 2023. Version 4.

[4] SeleniumHQ. Selenium webdriver, 2023.

[5] Ritchie van der Vegt. Polars: Fast dataframe library in rust and python, 2023.