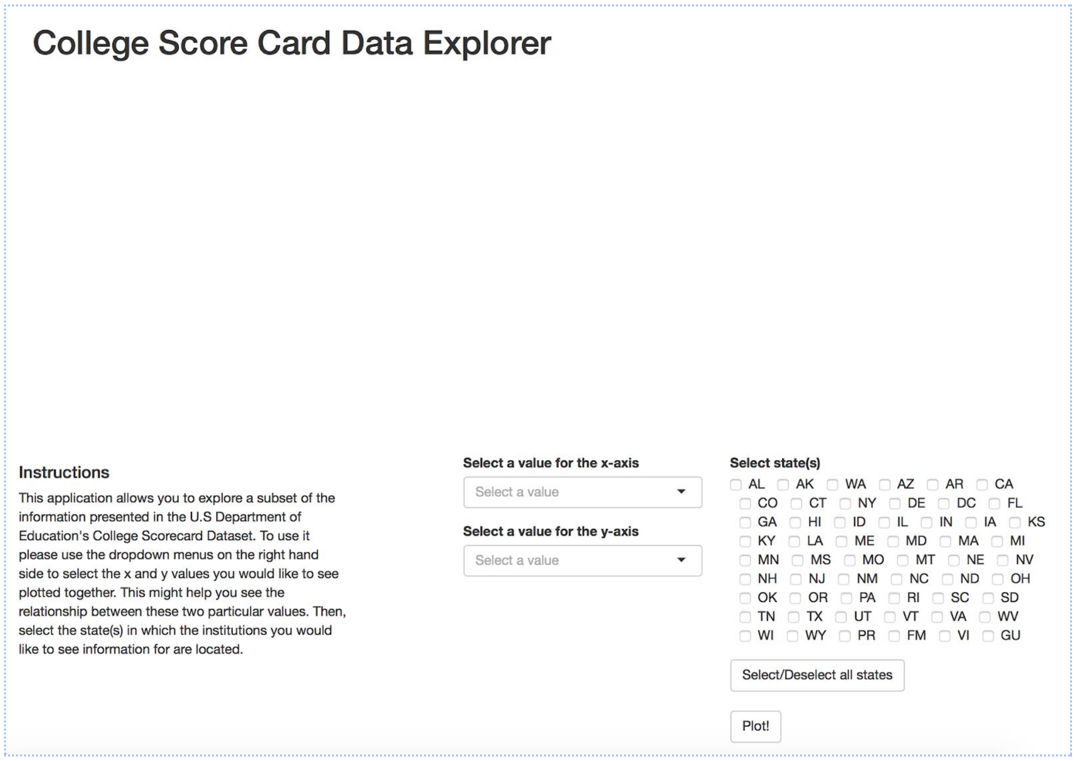
College Scorecard Data Explorer

Veronica Rivera
March 17, 2021

Introduction

Navigating the college application process can be extremely challenging for students and their families, especially for those who might have very little experience with this process, such as underrepresented and/or first generation students. I know that when I was applying to college, it was the first time my family and I were going through the process and we had so many questions and difficulty finding the answers. Recently I came across the U.S Department of Education's College Scorecard Dataset, which contains aggregate information for various institutions of higher education in the U.S such as enrollment information, costs, and admissions statistics for the 1996-97 academic year through 2018-19. When I first opened the dataset I was overwhelmed with the size and sheer amount of information contained in the dataset. Many of the columns were labelled in such a way that I was able to parse the information only after clicking through multiple links on the College Scorecard Dataset website and reading through dense documentation. Yet, the dataset contains information that could be highly relevant to college-bound students and their families and that could help them make decisions about which colleges to apply to and attend. It is important for this information to be more easily accessible and understood.

I wanted to create an interactive web application that could display some of the information from the College Scorecard dataset in a way that is easier to understand for someone who does not want to, or is not able to, dig through various links and pages of documentation. Since I was not able to talk to any college-bound students or their families prior to beginning work on an initial prototype, I developed the first version based on my own experiences. When I was applying to colleges my family and I were interested in the admissions rate and cost. So for this first prototype I thought it might be interesting to allow users to explore various relationships between variables related to admissions and cost across different institutions. I decided to create an interactive web application that allows users to create a scatterplot by selecting x and y values among four different variables related to admissions rate and cost: admissions rate, in-state tuition, out of state tuition, percent of students with federal loans, and percent of students with pell grant. The application also allows users to view data points only for certain states. The application is shown in Figures 1 and 2 below.



College Score Card Data Explorer

Instructions

This application allows you to explore a subset of the information presented in the U.S Department of Education's College Scorecard Dataset. To use it please use the dropdown menus on the right hand side to select the x and y values you would like to see plotted together. This might help you see the relationship between these two particular values. Then, select the state(s) in which the institutions you would like to see information for are located.

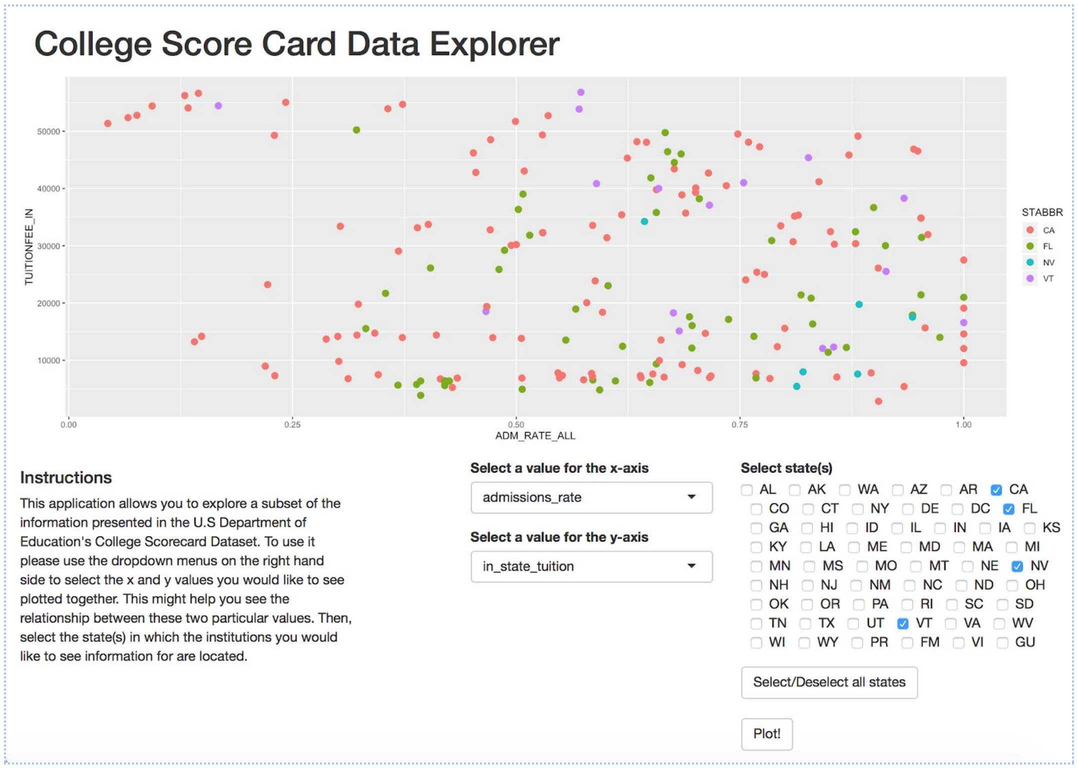
Select a value for the x-axis: Select a value

Select a value for the y-axis: Select a value

Select state(s)

☐ AL ☐ AK ☐ WA ☐ AZ ☐ AR ☐ CA
☐ CO ☐ CT ☐ NY ☐ DE ☐ DC ☐ FL
☐ GA ☐ HI ☐ ID ☐ IL ☐ IN ☐ IA ☐ KS
☐ KY ☐ LA ☐ ME ☐ MD ☐ MA ☐ MI
☐ MN ☐ MS ☐ MO ☐ MT ☐ NE ☐ NV
☐ NH ☐ NJ ☐ NM ☐ NC ☐ ND ☐ OH
☐ OK ☐ OR ☐ PA ☐ RI ☐ SC ☐ SD
☐ TN ☐ TX ☐ UT ☐ VT ☐ VA ☐ WV
☐ WI ☐ WY ☐ PR ☐ FM ☐ VI ☐ GU

Figure 1. Application when it is first run



College Score Card Data Explorer

Instructions

This application allows you to explore a subset of the information presented in the U.S Department of Education's College Scorecard Dataset. To use it please use the dropdown menus on the right hand side to select the x and y values you would like to see plotted together. This might help you see the relationship between these two particular values. Then, select the state(s) in which the institutions you would like to see information for are located.

Select a value for the x-axis: admissions_rate

Select a value for the y-axis: in_state_tuition

Select state(s)

☐ AL ☐ AK ☐ WA ☐ AZ ☐ AR ☑ CA
☐ CO ☐ CT ☐ NY ☐ DE ☐ DC ☑ FL
☐ GA ☐ HI ☐ ID ☐ IL ☐ IN ☐ IA ☐ KS
☐ KY ☐ LA ☐ ME ☐ MD ☐ MA ☐ MI
☐ MN ☐ MS ☐ MO ☐ MT ☐ NE ☑ NV
☐ NH ☐ NJ ☐ NM ☐ NC ☐ ND ☐ OH
☐ OK ☐ OR ☐ PA ☐ RI ☐ SC ☐ SD
☐ TN ☐ TX ☐ UT ☑ VT ☐ VA ☐ WV
☐ WI ☐ WY ☐ PR ☐ FM ☐ VI ☐ GU

Figure 2. Application in use. Displaying admissions rate vs. in-state tuition for institutions in CA, FL, NV, and VT.

Technical Details and Development Process

When creating the application, there were several factors that I needed to consider that ended up influencing my choice of libraries and programming language. Since I wanted the plot to update based on the values chosen by the user, handling reactivity was important. Another factor to consider was how the plot would be scaled. Since the application allows users to select several combinations of x and y values, I needed to consider upfront how I was going to handle appropriately scaling the axes differently within the same plot. Finally, I wanted my application to present information clearly, and have a clean UI. Since I wanted to create a web application, this meant that I also needed to be able to style both the plot and the interface that the user is going to interact with.

One of the first libraries that I considered using was D3, a javascript library for creating dynamic, interactive web-based visualizations. Since D3 is a javascript library, it would work well with HTML and CSS, allowing me to style the application as desired. However, one of the drawbacks of using D3 is the development time. In D3, I would have needed to create the entire scatterplot in my application from scratch, rather than using a built in function. And for the kind of plot that I wanted to create, it did not seem like I needed the entire functionality and power provided by D3. Another related drawback of using D3 for this project is that because I would have needed to implement the scatterplot from scratch, this also meant that I would have needed to manually handle the scaling of the axes on the different plots that result from the various combinations of user input. There is no easy way in D3 to automatically scale the axes. However, ggplot2 in R would provide me with this functionality. Since ggplot2 is a package within R, using it would allow me to easily build a scatterplot that I could then customize, rather than building one completely from scratch. In order to use ggplot2 and still create an interactive application, I decided to use Shiny. Shiny is an R package that would allow me to use all the functions and libraries within R, and since it also supports integration with HTML and CSS I would be able to style my application in much the same way as I would have with D3. Shiny not only supports the use of HTML and CSS, but it also has a built-in sub-library of interface builder functions that allow one to write HTML and CSS using R functions. Shiny also has a lot of built-in functions for UI elements (e.g. drop down menus) and out of the box styling, which made it so that I could style my application with less code.

Since the College Scorecard dataset is so large, the first thing I did upon downloading the CSV file from their site was clean up the data by removing rows that had missing values in the columns that I was interested in presenting to the user as x and y variables to choose from. This was relatively straightforward to do in R. In R missing values in a CSV are filled in with “NA”. In order to remove these values I used the `na.omit()` function on the data frame that contained the CSV file that was read in. The result was another data frame where incomplete rows had been deleted. Even though this meant that the resulting CSV file would now have missing institutions, for my goals in this project, this was fine since I would not have been able to present missing data to the user in the first place.

For my application I chose to create two dropdown UI elements for the user to select the x and y values to display in the scatterplot. One of the issues I ran into early on was that the plot was updating immediately after the user selected the first value, before they had a chance to select the second value. This was due to the way reactivity works in Shiny. Within Shiny, by default changes to an input object, such as a dropdown UI element, propagate to an output object, such as the scatterplot, immediately. To avoid this, I decided to add a button that the user could click when they were done selecting x and y values from the dropdown. In order for the plot to update when the user clicks the button, I delayed the reaction using the `isolate()` function within Shiny. The `isolate()` function allows an observer to access the value of a reactive value or expression, in this case the scatterplot, without taking a dependency on it. This allows the user to change the x and y values in the drop down menu as many times as they want without updating the graph automatically.

Another feature that I wanted to incorporate in my application is the ability for users to select certain states to display in the scatterplot. In order to do this I created a checkbox UI element using the `checkboxGroupInput()` function in Shiny. Shiny has another similar function, `checkboxInput()`. However, the `checkboxGroupInput()` function was better suited to my needs because it creates a group of checkboxes that can toggle multiple choices independently, while the `checkboxInput()` function can only toggle a single value. To add all the possible states corresponding to institutions in the College Scorecard dataset as checkbox values, on the backend I selected the column containing the states from the original dataframe, created a dataframe with all duplicates removed, and then passed that dataframe into the `checkboxGroupInput()` function as the list of values for the checkboxes. I also wanted an easy way for users to select/deselect all states, so I created a single button that would select/deselect all checkbox values. To implement this functionality I used the `updateCheckboxGroupInput()` function in Shiny, which observes the number of times the select/deselect button is clicked, and updates the checkboxes according to whether the number of times the button has been clicked is even or odd.

To create the scatterplot itself I used the ggplot2 library. Since I was working with R I had the option to use R's basic built in scatterplot function as well. However, I opted to use ggplot2 mostly because as mentioned previously I wanted the axes' scale to update automatically depending on the x and y values selected by the user. Additionally, the default graphs produced by ggplot2 are much nicer, which meant I would have to spend less time trying to customize the scatterplot to look appealing. Finally, I knew that I wanted the data points in the scatterplot for each state to be displayed in a different color. With ggplot2 I would be able to do this in a very straightforward way. One of the components of a ggplot2 graph are aesthetic mappings that describe how variables in the data are mapped to visual properties of data points called geoms. Therefore, in order to map each state to a different color, I was able to set the color component of the aesthetic mapping to the list of states. By doing it this way, ggplot automatically selects a different color for each state's datapoint, and creates a legend in the graph as well.

Conclusion

Currently my College Scorecard data explorer application is an early stage working prototype. In the future, I would like to continue refining it and adding more features that could best serve the needs of students and families applying to college. One feature I would like to add is allowing the user to hover over a data point in the graph to view more information about that institution (e.g. name, city, etc). For this prototype I chose which features and which x and y values to allow the user to interact with based on what my family and I would have wanted to know when I was applying to college. However, other individuals might have needs that I am not yet aware of. In the future I would be interested in talking with college-bound students and families to learn about what their needs for such an application would be, and potentially involving them in the design through participatory design approaches. On the technical side, developing this prototype has allowed me to see how powerful Shiny can be for creating these kinds of interactive web applications, so I intend to continue using it to further develop my app.