Q2 Force-directed graph layout

Technology	D3 Version 5 (included in the lib folder) Chrome v92.0 (or higher): the browser for grading your code Python http server (for local testing)
Allowed Libraries	D3 library is provided to you in the lib folder. You must NOT use any D3 libraries (d3*.js) other than the ones provided.
Max runtime	NA Deliverables
Delivrables	submission.(html/js/css): The HTML, JavaScript, CSS to render the graph. Do not include the D3 libraries or board_games.csv dataset.

You will experiment with many aspects of D3 for graph visualization. To help you get started, we have provided the submission.html file (in the Q2 folder) and an undirected graph dataset of boardgames, board_games.csv file (in the Q2 folder). The dataset for this question was inspired by a reddit post about visualizing boardgames as a network, where the author calculates the similarity between board games based on categories and game mechanics where the edge value between each board game (node) is the total weighted similarity index. This dataset has been modified and simplified for this question and does not fully represent actual data found from the post. The provided submission.html file will display a graph (network) in a web browser. The goal of this question is for you to experiment with the visual styling of this graph. Here is a helpful resource (about graph layout) for this question.

Note: You are welcome to split submission.html into submission.html, submission.css, and submission.js.

a. **Adding node labels**: Modify submission.html to show the node label (the node name, e.g., the source) at the top right of each node in **bold**. If a node is dragged, its label must move with it.
b. **Styling edges**: Style the edges based on the "value" field in the links array: • If the value of the edge is equal to 0 (similar), the edge should be gray, thick, and solid (The dashed line with zero gap is not considered as solid). • If the value of the edge is equal to 1 (not similar), the edge should be green, thin, and dashed.
c. **Scaling nodes**:

1. [1.5 points] Scale the radius of each node in the graph based on the degree of the node (you may try linear or squared scale, but you are not limited to these choices). Note: Regardless of which scale you decide to use, you should avoid extreme node sizes, which will likely lead to low-quality visualization (e.g., nodes that are mere points, barely visible, or of huge sizes). Note: D3 v5 does not support d.weight (which was the typical approach to obtain node degree in D3 v3). You may need to calculate node degrees yourself. Example relevant approach is here.
2. [1.5 points] The degree of each node should be represented by varying colors. Pick a meaningful color scheme (hint: color gradients). There should be at least 3 color gradations and it must be visually evident that the nodes with a higher degree use darker/deeper colors and the nodes with lower degrees use lighter colors. You can find example color gradients at Color Brewer.

d. [6 points] **Pinning nodes**:

1. [2 points] Modify the code so that dragging a node will fix (i.e., "pin") the node's position such that it will not be modified by the graph layout algorithm (Note: pinned nodes can be further dragged around by the user. Additionally, pinning a node should not affect the free movement of the other nodes). Node pinning is an effective interaction technique to help users spatially organize nodes during graph exploration. The D3 API for pinning nodes have evolved over time. We recommend reading this post when you work on this sub-question.
2. [1 points] Mark pinned nodes to visually distinguish them from unpinned nodes, i.e., show pinned nodes in a different color.
3. [3 points] Double clicking a pinned node should unpin (unfreeze) its position and unmark it. When a node is no longer pinned, it should move freely again.
e. [1 points] Add GT username: Add your Georgia Tech username (usually includes a mix of letters and numbers, e.g., gburdell3) to the top right corner of the force-directed graph (see example image). The GT username must be a element having the id: "credit"

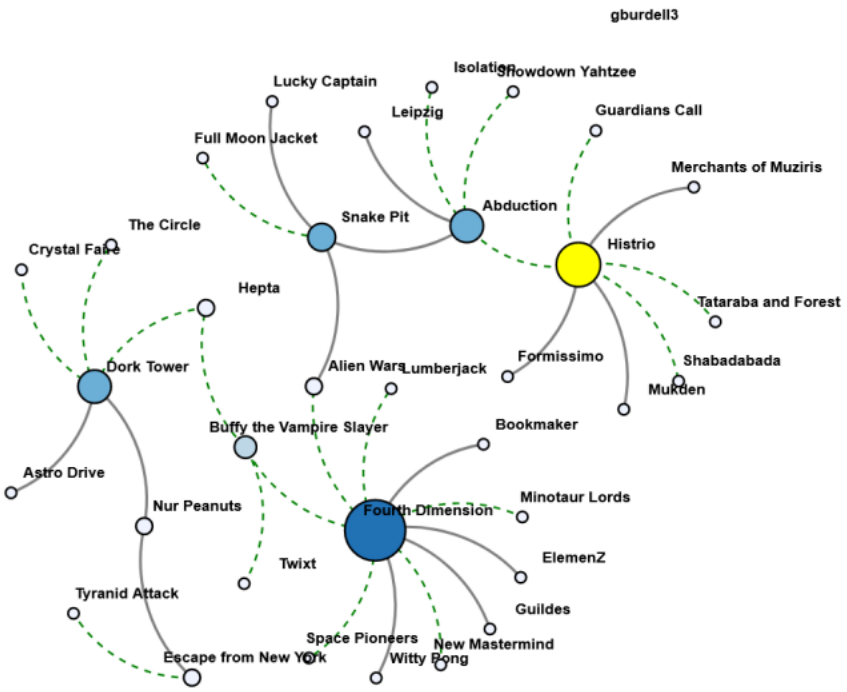


Figure 2a: Example of Visualization with pinned node (yellow). Your chart may appear different, and can earn full credit if it meets all the stated requirements.

Q3 Line Charts

Technology	D3 Version 5 (included in the lib folder) Chrome v92.0 (or higher): the browser for grading your code Python http server (for local testing)
Allowed Libraries	D3 library is provided to you in the lib folder. You must NOT use any D3 libraries (d3*.js) other than the ones provided.
Max runtime	NA
Deliverables	linecharts.(html / js / css): The HTML, JavaScript, CSS to render the line charts. Use the dataset provided in the file boardgame_ratings.csv (in the

	Q3 folder) to create line charts. Refer to the tutorial for line chart here or here

Note: You will create four charts in this question, which should be placed one after the other on a single HTML page, similar to the example image below (Figure 3). Note that your design need NOT be identical to the example; however, the submission must follow the DOM structure specified at the end of this question.

a. **Creating line chart**. Create a line chart (Figure 3a) that visualizes the number of board game ratings from November 2016 to August 2020 (inclusively), for the eight board games: [‘Catan’, 'Dominion', 'Codenames', 'Terraforming Mars', 'Gloomhaven', 'Magic: The Gathering', 'Dixit', 'Monopoly']. Use d3.schemeCategory10 to differentiate these board games. Add each board game’s name next to its corresponding line. For the x-axis, show a tick label for every three months. Use D3 axis.tickFormat() and d3.timeFormat() to format the ticks to display abbreviated months and years. For example, Jan 17, Apr 17, Jul 17. (See Figure 3a and its x-axis ticks).

- Chart title: Number of Ratings 2016-2020
- Horizontal axis label: Month. Use D3.scaleTime().
- Vertical axis label: Num of Ratings. Use a linear scale (for this part a).

b. **Adding board game rankings**. Create a line chart (Figure 3b) for this part (append to the same HTML page) whose design is a variant of what you have created in part a. Start with your chart from part a. Modify the code to visualize how the rankings of [‘Catan’, ‘Codenames’, ‘Terraforming Mars’, ‘Gloomhaven’] change over time by adding a symbol with the ranking text on their corresponding lines. Show the symbol for every three months, similar to the x-axis ticks in part a. (See Figure 3b). Add a legend to explain what this symbol represents next to your chart (See the Figure 3b bottom right).

- Chart title: Number of Ratings 2016-2020 with Rankings

c. **Axis scales in D3**. Create two line charts (Figure 3c-1,2) for this part (append to the same HTML page) to try out two axis scales in D3. Start with your chart from part b. Then modify the vertical axis scale for each chart: the first chart uses the square root scale for its vertical axis (only), and the second chart uses the log scale for its vertical axis (only). Keep the symbols and the symbol legend you implemented in part b. At the bottom right of the last chart, add your GT username (e.g., gburdell3, see Figure 3c-2 for example).

Note: the horizontal axes should be kept in linear scale, and only the vertical axes are affected.

Hint: You may need to carefully set the scale domain to handle the 0s in data.

■ First chart (Figure 3c-1)

- ○ Chart title: Number of Ratings 2016-2020 (Square root Scale)
- ○ This chart uses the square root scale for its vertical axis (only)
- ○ Other features should be the same as part b.

■ Second chart (Figure 3c-2)

- ○ Chart title: Number of Ratings 2016-2020 (Log Scale)

- This chart uses the log scale for its vertical axis (only)
- Other features should be the same as part b.

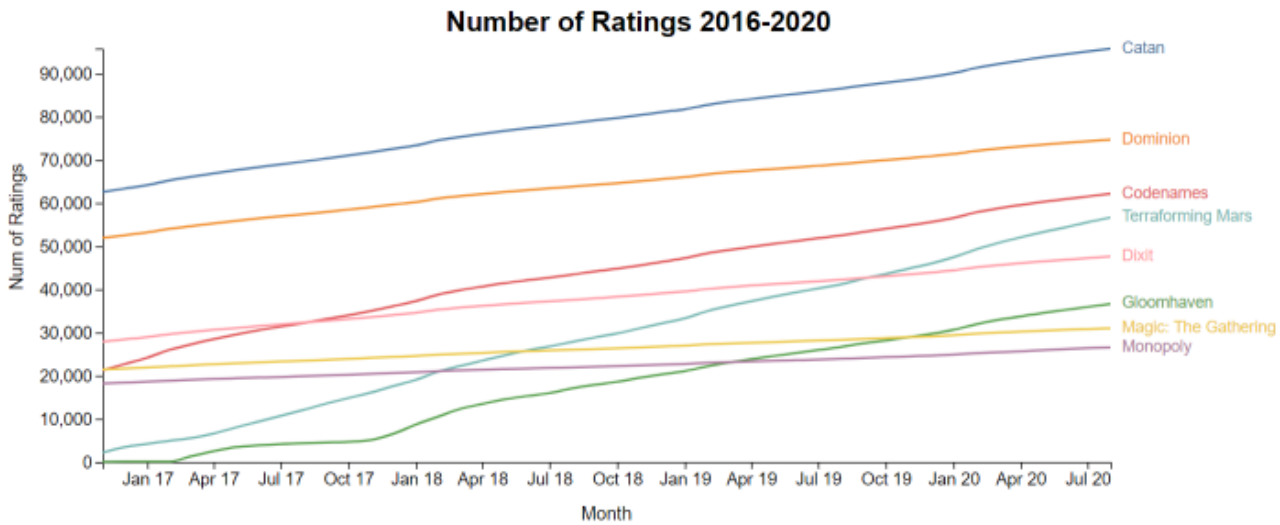


Figure 3a: Example line chart. Your chart may appear different, and can earn full credit if it meets all stated requirements.

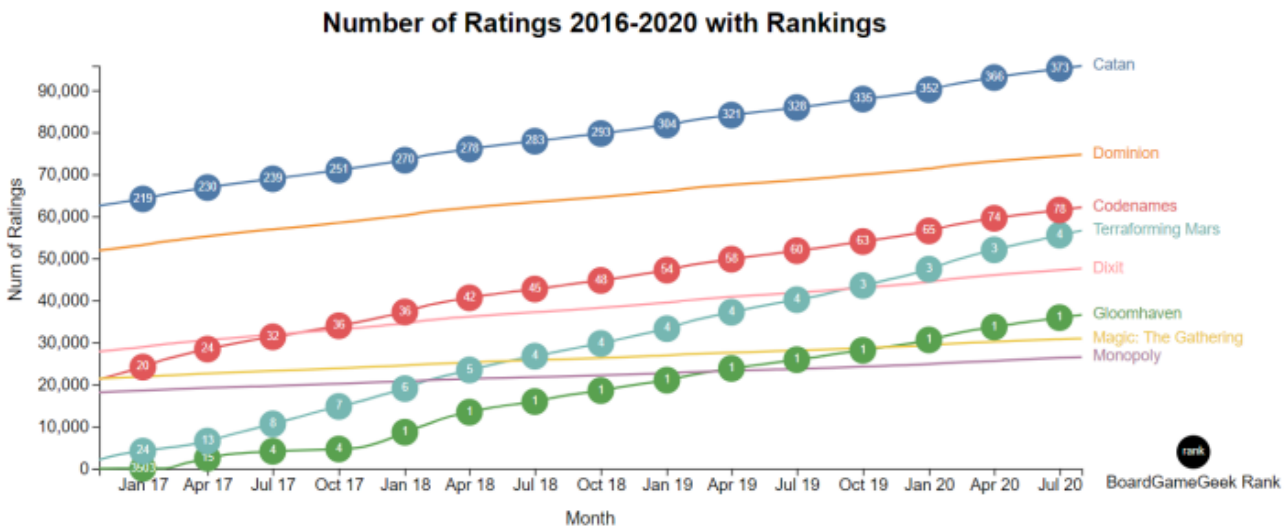


Figure 3b: Example of a line chart with rankings. Your chart may appear different, and can earn full credit if it meets all stated requirements.

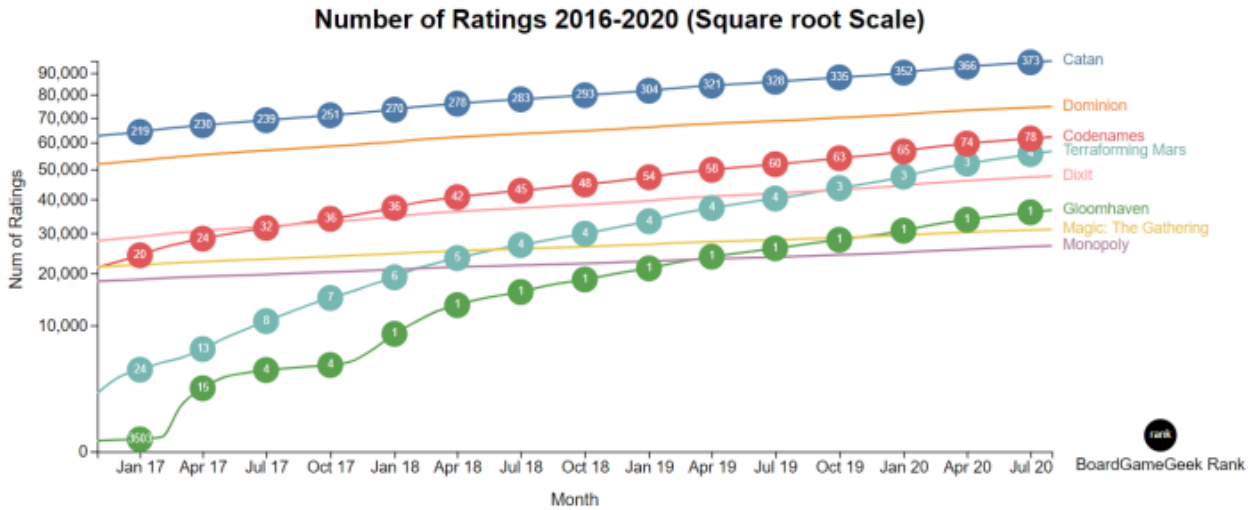


Figure 3c-1: Example of a line chart using square root scale. Your chart may appear different, and can earn full credit if it meets all stated requirements.

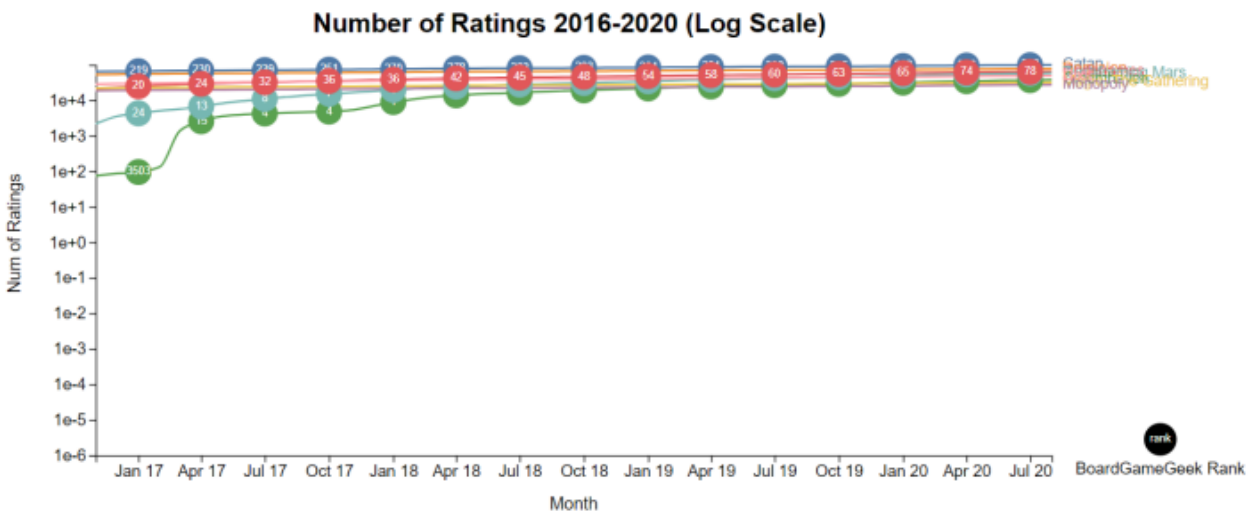


Figure 3c-2: Example of a line chart using log scale. Your chart may appear different, and can earn full credit if it meets all stated requirements.

Note: Your D3 visualization MUST produce the following DOM structure.

```
<svg id="svg-a"> plot (Q3.a)
|
+-- <text id="title-a"> chart title
|
+-- <g id="plot-a"> containing Q3.a plot elements
    |
    +-- <g id="lines-a"> containing plot lines, line labels
    |
    +-- <g id="x-axis-a"> x-axis
    |   |
    |   +-- (x-axis elements)
    |   |
    |   +-- <text> x-axis label
    |
```

```
    +-- <g id="y-axis-a"> y-axis
        |
        +-- (y-axis elements)
        |
        +-- <text> y-axis label

<svg id="svg-b"> plot (Q3.b)
|
+-- <text id="title-b"> chart title
|
+-- <g id="plot-b"> containing Q3.b plot elements
|   |
|   +-- <g id="lines-b"> containing plot lines, line labels
|   |
|   +-- <g id="x-axis-b"> for x-axis
|   |   |
|   |   +-- (x-axis elements)
|   |   |
|   |   +-- <text> x-axis label
|   |
|   +-- <g id="y-axis-b"> for y-axis
|   |   |
|   |   +-- (y-axis elements)
|   |   |
|   |   +-- <text> for y-axis label
|   |
|   +-- <g id="symbols-b"> containing plotted symbols, symbol labels
|
+-- <g id="legend-b"> containing legend symbol and legend text element(s)


<svg id="svg-c-1"> plot (Q3.c1): same as format for Q3.b, with c-1 in ids
(e.g., id="svg-c-1", etc.)

<svg id="svg-c-2"> plot (Q3.c2): same as format for Q3.b, with c-2 in ids
(e.g., id="svg-c-2", etc.)

<div id="signature"> containing GT username
```

Q4 Interactive Visualization

Technology	D3 Version 5 (included in the lib folder) Chrome v92.0 (or higher): the browser for grading your code Python http server (for local testing)
Allowed Libraries	D3 library is provided to you in the lib folder. You must NOT use any D3 libraries (d3*.js) other than the ones provided.
Max runtime	NA
Deliverables	interactive.(html/js/css): The HTML, JavaScript, CSS to render the visualization in Q4. Do not include the D3 libraries or average-rating.csv dataset.

Use the dataset average-rating.csv provided in the Q4 folder to create an interactive frequency polygon line chart. This dataset contains a list of games, their ratings and supporting information like the numbers of users who rated a game and the year a game was published. In the data sample below, each row under the header represents a game name, year of publication, average

rating, and the number of users who rated the game. Helpful resource to work with nested data in D3: http://bl.ocks.org/phoebebright/raw/3176159/

name,year,average_rating,users_rated
Codenames,2015,7.71148,51209
King of Tokyo,2011,7.23048,48611

a. **Create a line chart**. Summarize the data by displaying the count of board games by rating for each year. Round each rating down to the nearest integer, using Math.floor(). For example, a rating of 7.71148 becomes 7. For each year, sum the count of board games by rating. Display one plot line for each of the 5 years (2015-2019) in the dataset. Note: the dataset comprises year data from 2011 to 2020; this question asks to plot lines for the years 2015-2019. If some of the datapoints in the chart do not have ratings, generate dummy values (0s) to be displayed on the chart for the required years. All axes must start at 0, and their upper limits must be automatically adjusted based on the data. Do not hard-code the upper limits.

- The vertical axis represents the count of board games for a given rating. **Use a linear scale.**
- The horizontal axis represents the ratings. Use a linear scale.

b. **Line styling, legend, title and username.**

- For each line, use a different color of your choosing. Display a filled circle for each rating-count data point.
- Display a legend on the right-hand portion of the chart to show how line colors map to years.
- Display the title "Board games by Rating 2015-2019" at the top of the chart.
- Add your GT username (usually includes a mix of lowercase letters and numbers, e.g., gburdell3) beneath the title (see example figure 4b).

Figure 4b shows an example line chart design. Yours may look different, but can earn full credit if it meets all stated requirements.

Note: The data provided in average-rating.csv requires some processing for aggregation. **All aggregation must only be performed in JavaScript; you must NOT modify average-rating.csv.** That is, your code should first read the data from .csv file as is, then you may process the loaded data using JavaScript.

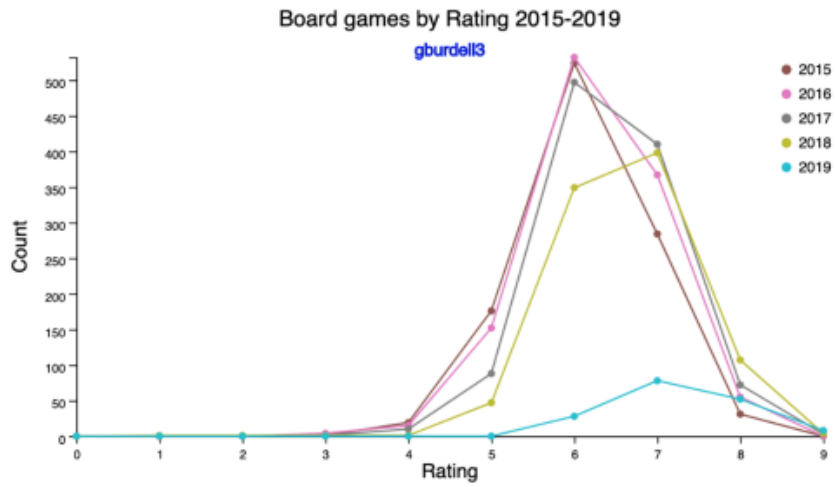


Figure 4b: Line chart representing count of board games by rating for each year. Your chart may appear different, but can earn full credit if it meets all stated requirements.

Interactivity and sub-chart. In the next few sub-questions, you will create event handlers to detect mouseover and mouseout events over each circle that you added in Q4.b.

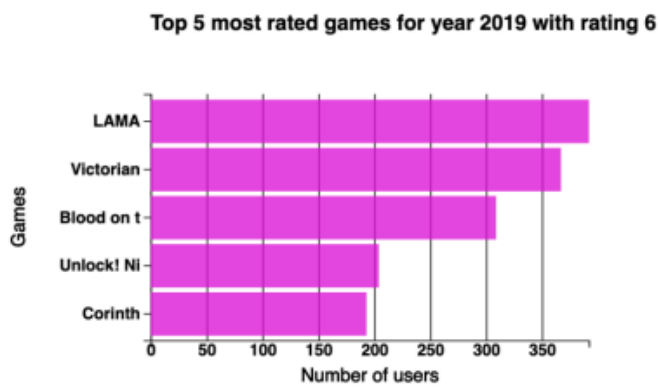


Figure 4c: Bar chart representing the number of users who rated the top 5 board games with the rating 6 in year 2019. Your chart may appear different, but can earn full credit if it meets all stated requirements.

c. **Create a horizontal bar chart**, so that when hovering over a circle, that bar chart will be shown below the line chart. The bar chart displays the top 5 board games that received the highest numbers of user ratings (users_rated), for the hovered year and rating. For example, hovering over the rating-6 circle for 2019 will display the bar chart for the number of users who rated the top 5 board games. If a certain year/rating combination has fewer than 5 entries, it should display as many as there are. Figure 4c shows an example design. Show one bar per game. The bar length represents the number of users who rated the game.

Note: No bar chart should be displayed when the count of games is 0 for hovered year and rating. Axes: All axes should be automatically adjusted based on the data. Do not hard-code any values.

• The vertical axis represents the board games. Sort the game names in ascending order, such that the game with the smallest users_rated is at the bottom, and the game with the highest users_rated is at the top. Some boardgame names are quite long. For each game name, display its first 10 characters (if a name has fewer than 10 characters, display them all). A space counts as a character. The horizontal axis represents the number of users who rated the game (for the hovered year and rating). Use a linear scale.

• Set horizontal axis label to 'Number of users' and vertical axis label to 'Games'.

d. **Bar styling, grid lines and title**

• Bars: All bars should have the same color regardless of year or rating. All bars for the specific year should have a uniform bar thickness.

• Grid lines should be displayed.

• Title: Display a title with the format "Top 5 Most Rated Games of with Rating " at the top of the chart where and are what the user hovers over in the line chart. For example, hovering over rating 6 in 2015, the title would read: "Top 5 Most Rated Games of 2015 with Rating 6"

e. **Mouseover Event Handling**

• The bar chart and its title should only be displayed during mouseover events for a circle in the line chart.

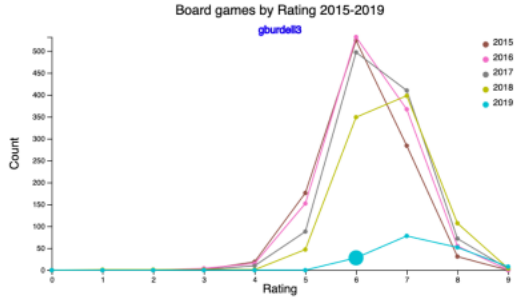
• The circle in the line chart should change to a larger size during mouseover to emphasize that it is the selected point.

• When count of games is 0 for hovered year and rating, no bar chart should be displayed. The hovered-over circle on the line graph should still change to a larger size to show it is selected.

f. **Mouseout Event Handling**

• The bar chart and its title should be hidden from view on mouseout and the circle previously mouseover-ed should return to its original size in the line chart.

The graph should exhibit interactivity similar to Figure 4f where the mouse is over the larger circle.



Top 5 most rated games for year 2019 with rating 6

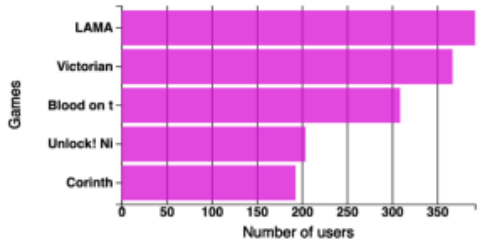


Figure 4f: Line chart and bar chart demonstrating interactivity. Your chart may appear different, but can earn full credit if it meets all stated requirements.

Note: Your D3 visualization MUST produce the following DOM structure.

```
<svg> containing line chart
|
+-- <g id="lines"> element containing all line elements
|     |
|     +-- <path> elements for plotted lines
|     |
|     +-- <text> elements for plotted lines
|
+-- <g id="x-axis-lines"> element for x-axis
|
+-- <g id="y-axis-lines"> element for y-axis
|
+-- <g id="circles"> element for all circular elements
|     |
|     +-- <circle> elements
|
+-- <g id="line_chart_title"> element for line chart title
|     |
|     +-- <text> element for line chart title
|
+-- <g id="credit"> element for GT username
|     |
|     +-- <text> element for GT username
|
+-- <g id="legend"> element for legend


|     |
|     +-- (<circle> elements for legend)
|     |
|     +-- (<text> elements for legend)
|
+-- <text> element for x axis label
|
+-- <text> element for y axis label
```

```
<svg id="barchart" > containing bar chart
|
+-- <g id="bars"> element for bars
|     |
|     +-- <rect> elements for bars
|
+-- <g id="x-axis-bars"> element for x-axis
|
+-- <g id="y-axis-bars"> element for y-axis
|
+-- <g id="circles"> element for all circular elements
|     |
|     +-- <circle> elements
|
+-- <g id="bar_chart_title"> element for bar chart title
|     |
|     +-- <text> element for bar chart title
|
+-- <g id="bar_x_axis_label"> element for x axis label
|     |
|     +-- <text> elements for x axis label
|
+-- <g id="bar_y_axis_label"> element for y axis label
|
+-- <text> elements for y axis label
```

Q5 Choropleth Map of Board Game Ratings

Technology	D3 Version 5 (included in the lib folder) Chrome v92.0 (or higher): the browser for grading your code Python http server (for local testing)
Allowed Libraries	s D3 library is provided to you in the lib folder. You must NOT use any D3 libraries (d3*.js) other than the ones provided. On Gradescope, these libraries are provided for you in the auto-grading environment.
Max runtime	NA
Deliverables	choropleth.(html/js/css): Modified file(s) containing all html, javascript, and any css code required to produce the plot

Choropleth maps are a very common visualization in which different geographic areas are colored based on the value of a variable for each geographic area. You have most probably seen choropleth maps showing quantities like unemployment rates for each county in the US, or the number of confirmed COVID-19 cases per 10,000 people at the county level.
We will use choropleth maps to examine the popularity of different board games across the world. We have provided two files in the Q5 folder, ratings-by-country.csv and world_countries.json.

- Each row in ratings-by-country.csv represents about a game's information for a country, in the form of , where
  - Game: the name of a game, e.g., Catan.
  - Country: a country in the world, e.g., United States of America.
  - Number of Users: the number of users who have rated Game who are from Country.
  - Average Rating: the mean rating given to Game by users who are from Country.

  This dataset has been preprocessed and filtered to include only those games that have been rated by more than 1000 users in the world.
- The world_countries.json file is a geoJSON, containing a single geometry collection: countries. You can find examples of map generation using geoJSON here and here.

LINKS BELOW ARE NOT ACTIVE. I WILL PROVIDE THE ORIGINAL DOCUMENT LATER

a. **[20 points]** Create a choropleth map using the provided data, and use Figure 5a and 5b as references.

1. **[5 points]** Dropdown lists are commonly used on dashboards to enable filtering of data. Create a dropdown list (see example in Figure 5a) to allow users to select which game's data are displayed.
   - The list options should be obtained from the `Game` column of the csv file.
   - Sort the list options in alphabetical order. Set the default display value to the first option.
   - Selecting a different game from the dropdown list should update both the choropleth map (see part 2) and the legend (see part 3) accordingly.

2. **[10 points]** Load the data from *ratings-by-country.csv* and create a choropleth map such that the color of each country in the map corresponds to the **average rating of the game selected in the dropdown** in each country. Use a "Natural Earth" projection of the geoJSON file. Name the function for calculating path as 'path', to help the auto-grader locate it.

   Promise.all() is provided within the skeleton code and you can use it to read in both the world json file and game data csv file. Example usage can be seen here.

   Many countries have no ratings for some games - these should be **colored grey**.

   For those countries that do have ratings for the selected game, use a quantile scale to generate the color scheme based on the average rating by country. Color them along a gradient of exactly 4 gradations from a single hue, with darker colors corresponding to higher rating values and lighter colors correspond to lower values (see gradient examples at Color Brewer).

   About Scaling Colormaps: In order to create effective visualizations that highlight patterns of interest, it is important to carefully think about the relationship between the range and distribution of values being displayed (the domain) and the color scale the values are mapped to (the range). Many types of mapping functions are possible, e.g., we could use a linear mapping where the lowest game rating is mapped to the first value in the color scheme, the highest game rating is mapped to the highest value in the color scheme, and intermediate ratings are mapped to hues in the middle. This article illustrates the value of choosing appropriate endpoints for linear color maps, or log-scaling the domain so that large but relatively infrequent values do not cause differences between smaller values to be washed out. In our case, most board games have similar average ratings across countries, e.g. Catan has an average rating close to 9.3 in almost all countries, making it challenging to perceive relative differences in popularity. To address this, we can compute quantiles of the domain data—game rating values that divide the ordered list of average ratings per country into roughly equally-sized groups. Here, we will get 4 groups, a special case of quantiles called "quartiles" since the data are divided into quarters.

Hint: You can verify the correctness of the quartiles generated by using the 'quartile' function in Excel. Open ratings-by-country.csv and filter the data for one game (say Catan). Then use the quartile function to get the 0th, 1st, 2nd, 3rd and 4th quartile values from the Average Rating column. Here [0th quartile, 1st quartile], [1st quartile, 2nd quartile], [2nd quartile, 3rd quartile], [3rd quartile, 4th quartile] will represent the 4 groups of values generated by the d3 quantile scale. Use all the countries listed in ratings-by-country.csv to generate your quartiles, including ones which may not appear in the geoJSON.

3. **[5 points]** Add a **vertical** legend showing how colors map to the average rating for a particular game. The legend must update for the quartiles of the selected game, and display values formatted to show precision up to 2 decimal places. You must use **exactly 4** color gradations in your submission. It is recommended, but not required, to use d3-legend.min.js (in the **lib** folder) to create the legend for the scale you use. Also, **display your GT username** (e.g., gburdell3) beneath the map.

b. **[5 points]** Add a tooltip using the d3-tip.min library (in the **lib** folder). On hovering over a country, the tooltip should show the following information on separate lines:

- *Country* name
- *Game*
- *Avg Rating* for that game in that country (Figure 5b demonstrates this for Catan)
- *Number of Users* from the country who rated the game

For countries with no data, the tooltip should display "N/A" for *Avg Rating* and *Number of Users*.

The tooltip should appear when the **mouse** hovers over the country. On mouseout, the tooltip should disappear. You can position the tooltip a small distance away from the **mouse cursor and have it follow the cursor**, which will prevent the tooltip from "flickering" as you move the mouse around quickly (the tooltip disappears when your mouse leaves a state and enters the tooltip's bounding box). Alternatively, you may position the tooltip at a location (picked by you) such that it is close to the country the cursor is currently at. Please ensure that the tooltip is fully visible (i.e., not clipped, especially near the page edges). If the tooltip becomes clipped you may lose points.

Note: You **must** create the tooltip by **only** using **d3-tip.min.js** in the **lib** folder.

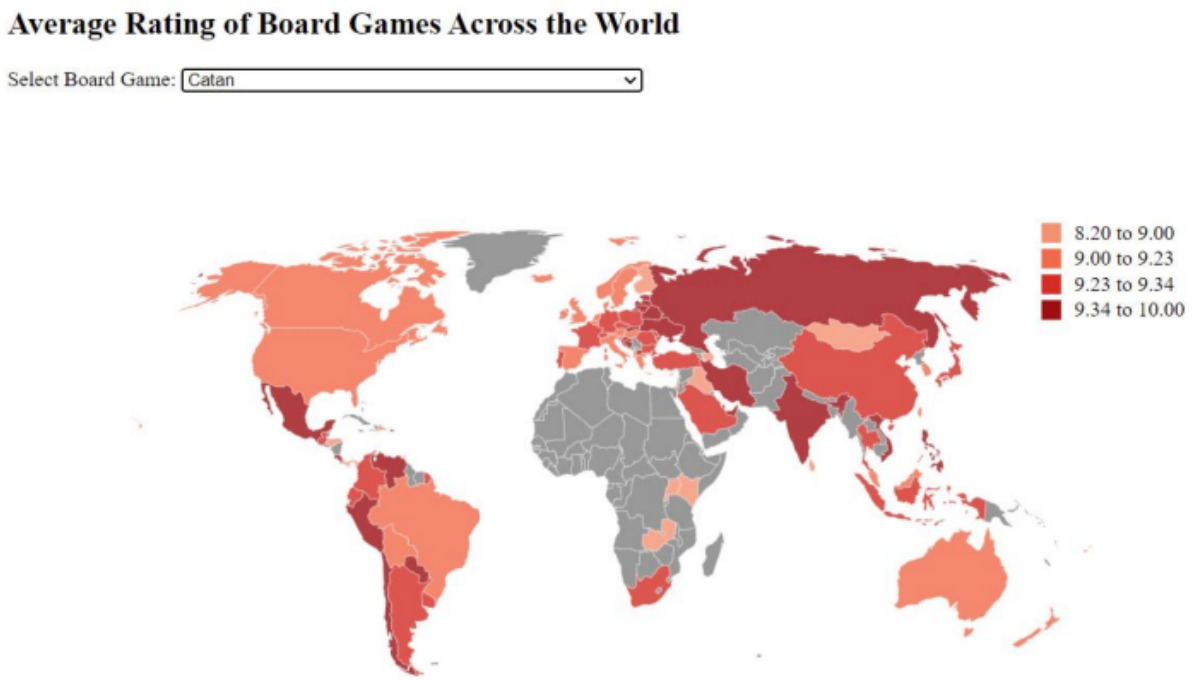


Figure 5a: Reference example for Choropleth Map showing average rating of Catan. Your chart may appear different and earn full credit as long as it meets all stated requirements.

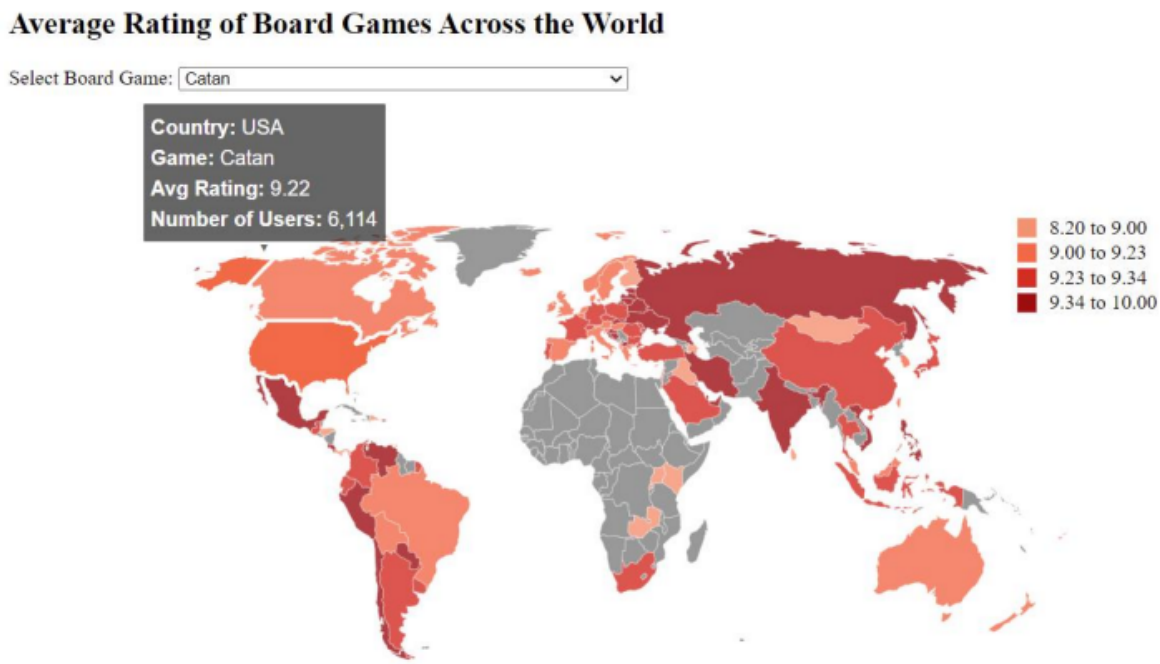


Figure 5b: Reference example for Choropleth Map showing tooltip. Your chart may appear different and earn full credit as long as it meets all stated requirements.

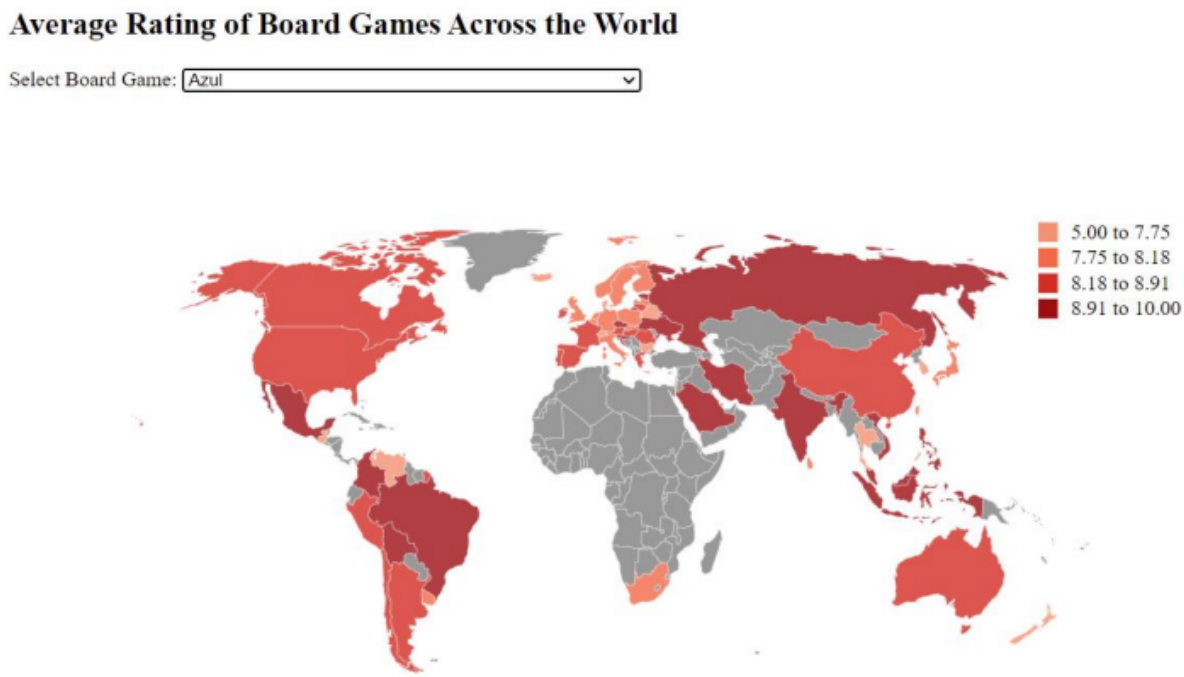


Figure 5c: Reference example showing updated Choropleth and legend for Azul. Your chart may appear different and earn full credit as long as it meets all stated requirements.

Tips and hints

- Countries without data should be colored grey. These countries can be found using a condition that compares the country's average rating with 'undefined'.
- It is optional for your visualization to show (or not show) Antarctica.
- D3-tip warning may be ignored if it does not break the code.
- You may consider clearing the SVG and create a new map when selecting a new game.

Note: You may change the given code in choropleth.html as necessary. Your D3 visualization **MUST** produce the following DOM structure.

```
<select id="gameDropdown"> sorted list of game options
|
+-- <option> (one option for each game; value = game name)
    |
    +-- text (= game name)

<svg> contains choropleth map
|
+-- <g id="countries">
|   |
|   +-- <path>s for each country
|
+-- <g id="legend"> legend

<div id="tooltip"> contains tooltip to display (Q5.b)
|
+-- (text for tooltip)
```