CSE 6242/CX 4242: Data and Visual Analytics | Georgia Tech | Spring 2018

Homework 2: D3 Graphs and Visualization

Due: Wednesday, February 28, 2018, 11:55 PM EST

Prepared by Arathi Arivayutham, Siddharth Gulati, Jennifer Ma, Mansi Mathur, Vineet Vinayak Pasupulety, Neetha Ravishankar, Polo Chau

Submission Instructions and Important Notes:

It is important that you read the following instructions carefully and also those about the deliverables at the end of each question or you may lose points.

- ❏ Always check to make sure you are using the most up-to-date assignment PDF (e.g., re-download it from the course homepage if unsure).
- ❏ Submit a single zipped file, called "HW2-{YOUR_LAST_NAME}-{YOUR_FIRST_NAME}.zip", containing all the deliverables including source code/scripts, data files, and readme. Example: 'HW2-Doe-John.zip' if your name is John Doe. **Only .zip is allowed** (no other format will be accepted).
- ❏ You may collaborate with other students on this assignment, but you must write your own code and give the explanations in your own words, and also mention the collaborators' names on T-Square's submission page. All GT students must observe the honor code. **Suspected plagiarism and academic misconduct will be reported to and directly handled** by the Office of Student Integrity (OSI). Here are some examples similar to Prof. Jacob Eisenstein's NLP course page (grading policy):
    - ❏ **OK:** discuss concepts and strategies (e.g., how cross-validation works, use hashmap instead of array)
    - ❏ **Not OK:** several students work on one master copy together (e.g., by dividing it up), sharing solutions, or using solution from previous years or from the web.
- ❏ If you use any "*slip days*", you must write down the number of days used in the T-square submission page. For example, "Slip days used: 1". Each slip day equals 24 hours. E.g., if a submission is late for 30 hours, that counts as 2 slip days.
- ❏ At the end of this assignment, we have specified a folder structure about how to organize your files in a single zipped file. 5 points will be deducted for not following this strictly.
- ❏ We will use auto-grading scripts to grade some of your deliverables (there are hundreds of students), so it is extremely important that you strictly follow our requirements. Marks may be deducted if our grading scripts cannot execute on your deliverables.
- ❏ Wherever you are asked to write down an explanation for the task you perform, stay within the word limit or you may lose points.
- ❏ In your final zip file, please do not include any intermediate files you may have generated to work on the task, unless your script is absolutely dependent on it to get the final result (which it ideally should not be).
- ❏ After all slip days are used, 5% deduction for every 24 hours of delay. (e.g., 5 pts for 100-point homework)
- ❏ We will not consider late submission of any missing parts of a homework assignment or project deliverable. To make sure you have submitted everything, download your submitted files to double check.

Grading

The maximum possible score for this homework is 115 points. Students in the undergraduate section (CX4242) can choose to complete any 100 points worth of work to receive the full 15% of the final course grade. For example, if a CX4242 student scores 115 pts, that student will receive (115 / 100) * 15 = 17.25 points towards the final course grade. To receive the full 15% score, students in the CSE6242 sections will need to complete all 115 points.

Important Prerequisites

Download the HW2 Skeleton that contains files you will use in this homework.

We highly recommend that you use the latest Firefox browser to complete this homework. We will grade your work using **Firefox 58.0.1 (or newer)**.

For this homework, you will work with **version 3 of D3**, provided to you in the **lib** folder. You must NOT use any other d3 libraries (d3*.js) other than the ones provided.

You may need to setup an HTTP server to run your D3 visualizations (depending on which web browser you are using, as discussed in the D3 lecture). The easiest way is to use SimpleHTTPServer in Python (for Python version 2.x). **You should run your local HTTP server in the root (hw2-skeleton) folder.**

All d3*.js files in the **lib** folder must be referenced using relative paths, e.g., **"../lib/<filename>"** in your html files (e.g., those in folders Q2, Q3, etc.). For example, suppose the file "Q2/graph.html" uses d3, its header should contain:

```
<script  type="text/javascript" src="../lib/d3.v3.min.js"></script>
```

It is incorrect to use an absolute path such as:

```
<script type="text/javascript" src="http://d3js.org/d3.v3.min.js"></script>
```

All questions that require reading from a dataset require you to submit the dataset in the deliverables too. In your html/js code**,** use a **relative path** to read in the dataset file**.** For example, since Q4 requires reading data from the `heatmap.csv` file (which should be submitted as part of the deliverables in the Q4 folder), the path should simply be 'heatmap.csv' and NOT an absolute path such as "C:/Users/polo/HW2-skeleton/Q4/heatmap.csv".

You can and are encouraged to decouple the style, functionality and markup in the code for each question. That is, you can use separate files for css, javascript and html.

Q1 [10 pts] Designing a Good Table and Visualizing Data with Tableau

songs.csv is a dataset of 648 songs that were part of the Eurovision Song Contest over the years 1998--2012.[1]. The description of the columns is available in **song_attribute_description.txt**.

a. **[5 pts] Good table design.** Create a table to display the details of the finalists in the competition in the year 2007. You can use any tool (e.g., Excel, HTML) to create the table. You should use only the following attributes for table creation: *Artist, Song, Place, Duration, Artist.gender, Energy and Happiness* You may reorder the columns while creating the table. Keep suggestions from class in mind when designing your table (see lectures slides, specifically slide #43 "How to fix the defaults", for what to try, but you are not limited to the techniques described).
   - ○ Describe your reason for choosing the techniques you use in **explanation_1a.txt** in no more than 50 words.
   - ○ **Save the table as table.(png/pdf).**

b. **[5 pts] Tableau:** In the form of a single chart consisting of only horizontal bars, visualize 5 acoustic attributes (Acousticness, Danceability, Liveness, Speechiness and Valence) for the top 4 songs that were played in the finals of 2007. Do ensure that the bars are grouped by the acoustic attributes (so it will be easier to compare an attribute's values across songs) and the axes are well-labeled.
   - ○ Explain your choices of techniques in **explanation_1b.txt**, using no more than 50 words.
   - ○ **Save the chart as song_characteristics.(png/pdf).**

   Our main goal here is for you to try out Tableau, a popular information visualization tool. We will accept most designs from you all. Thus, we have kept this part more open-ended, so you can practice making design decisions, which make the bar chart look aesthetically pleasing and informative.

   Tableau has provided us with student licenses. Go to tableau activation and select "Get Started". On the form, enter your Georgia Tech email address for "Business email" and "Georgia Institute of Technology" for "Organization". The Desktop Key for activation is available in T-Square Resources as "Tableau Desktop Key". This key is for your use in this course only. **Do not share the key with anyone.**

Q1 Deliverables:

The directory structure should be as follows:

```
Q1/
   table.(png / pdf)
   song_characteristics.(png / pdf)
   explanation_1a.txt
   explanation_1b.txt
   songs.csv
   song_attribute_description.txt
```

[1]: http://eschome.net/

- **table.(png / pdf)** - An image/screenshot of the table in Q1.a (png or pdf format **only**).
- **song_characteristics.(png / pdf)** - An image of the chart in Q1.b (png or pdf format **only**, Tableau workbooks will not be graded!). The image should **not** be an exact copy of the example visualization. The image should be clear and of high-quality.
- **explanation_1a.txt** - Your explanation for part Q1.a (maximum 50 words).
- **explanation_1b.txt** - Your explanation for part Q1.b (maximum 50 words).
- **songs.csv** and **song_attribute_description.txt** - The dataset and its description.

Q2 [15 pts] Force-directed graph layout

You will experiment with many aspects of D3 for graph visualization. To help you get started, we have provided the graph.html file (in the Q2 folder). **Note:** You are welcome to split the file graph.html into 3 files: graph.html, graph.css, and graph.js.

a. **[3 pts] Adding node labels**: Modify graph.html to show a node label (the node *name*, i.e., the *source*) to the right of each node. If a node is dragged, its label must also move with the node.

b. **[3 pts] Coloring links**: Color the links based on the “value” field in the links array. Assign the following colors:

If the value of the edge is < 3.0 : assign Red color to the link.
If the value of the edge is >= 3.0 and <= 4.0 : assign Blue color to the link.
If the value of the edge is > 4.0 : assign Green color to the link.

c. **[3 pts] Scaling node sizes:**

1. Scale the radius of each node in the graph based on the degree of the node.
2. In **explanation.txt**, using no more than 40 words, discuss your scaling method you have used and explain why you think it is a good choice. There are many possible ways to scale, e.g., scale the radii linearly, by the square root of the degree, etc.

d. **[6 pts] Pinning nodes** (fixing node positions):

1. Modify the html so that when you double click a node, it pins the node’s position such that it will not be modified by the graph layout algorithm (note: pinned nodes can still be dragged around by the user but they will remain at their positions otherwise). Node pinning is an effective interaction technique to help users spatially organize nodes during graph exploration.
2. Mark pinned nodes to visually distinguish them from unpinned nodes, e.g. pinned nodes are shown in a different color, border thickness or visually annotated with an “asterisk” (*), etc.
3. Double clicking a pinned node should unpin (unfreeze) its position and unmark it.

Q2 Deliverables:
The directory structure should be as follows:

```
                    Q2/
                        graph.html
                        graph.js, graph.css (if not included in graph.html)
                        explanation.txt
```

- **graph.html** - the html file created.
- **graph.(js / css)** - the js / css files if not included in graph.html
- **explanation.txt** - the text file explaining your choice of scaling method for Q2.c.

Q3 [15 pts] Scatter plots

Use the dataset provided in the file *wine_quality.csv* (in the folder Q3) to create a scatter plot.

Data used in this question was derived from https://archive.ics.uci.edu/ml/datasets/wine+quality.
Note: Please use ONLY the data provided in Q3 folder and NOT from the website as datasets have been modified for this question.

Refer to the tutorial for scatter plot here.

Quality of red wine based on its chemical composition is studied through various physicochemical tests and the data is recorded in the wine_quality dataset.
Attributes in the dataset:
Feature 1: Residual Sugar (g/cm3)
Feature 2: Chlorides (g/cm3)
Feature 3: Free Sulfur dioxide (g/cm3)
Feature 4: Total Sulfur dioxide (g/cm3)
Feature 5: Density (g/cm3)
Feature 6: pH
Feature 7: Sulphates (g/cm3)
Feature 8: Alcohol (% by volume)
Quality: Score between 0 to 10 **(Score >= 6 is considered to be of "Good" quality wine and score < 6 is considered to be of "Bad" quality wine)**

A. **[6 pts] Create two scatter plots**, one for each feature combination specified below. The dimensions of the canvas on which each plot is rendered should be 800x600 (width x height). In the scatter plots, visualize "Good" wine quality instances as blue crosses, and "Bad" instances as red circles. Both the crosses and circles must be hollow symbols. Add a legend to the top right corner showing how symbols map to the classes.
Plot 1 (Y-Axis vs. X-Axis): **pH (Features 6) vs. Alcohol (Feature 8)**
Plot 2 (Y-Axis vs. X-Axis)**: pH (Features 6) vs. Residual Sugar (Feature 1)**

Your scatter plots should be placed one below the other **on a single HTML page**, similar to the example image below (Figure 3)

Note: The example is meant to be used only as a reference for positions of the graphs. Your design need not be identical to the example.

Based on the scatter plot created for Feature 6 and Feature 8 (pH vs. Alcohol), create new plots for the following questions:

B. **[4 pts] Scaling symbol sizes.** Set the size of each symbol in the plot to be **inversely proportional (i.e., the larger the product, the smaller the symbol; the smaller the product, the larger the size of the symbol)** to the product of alcohol and pH. **Create a new scatter plot for this part** (append to the HTML page). Set the scaling coefficient properly to make the scatter plot legible.

Plot 3 (Y-Axis vs. X-Axis): **pH (Feature 6) vs. Alcohol (Feature 8)**

C. **[5 pts] Axis scales in D3.** Create two scatter plots for this part (append to the HTML page). Both scatter plots are created for the following feature combination

Plots 4 & 5 (Y-Axis vs. X-Axis)**: Sulphates (Feature 7) vs. pH (Feature 6)**

Use two different axis scales to create the scatter plots - the first plot uses the square root scale for its y-axis (only), and the second plot uses the log scale for its y-axis (only). Note that the x-axes (pH) should be kept in linear scale, and only the y-axes (Sulphates) are affected. Explain in no more than 50 words, in **explanation.txt**, when we may want to use square root scale and log scale in charts.

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

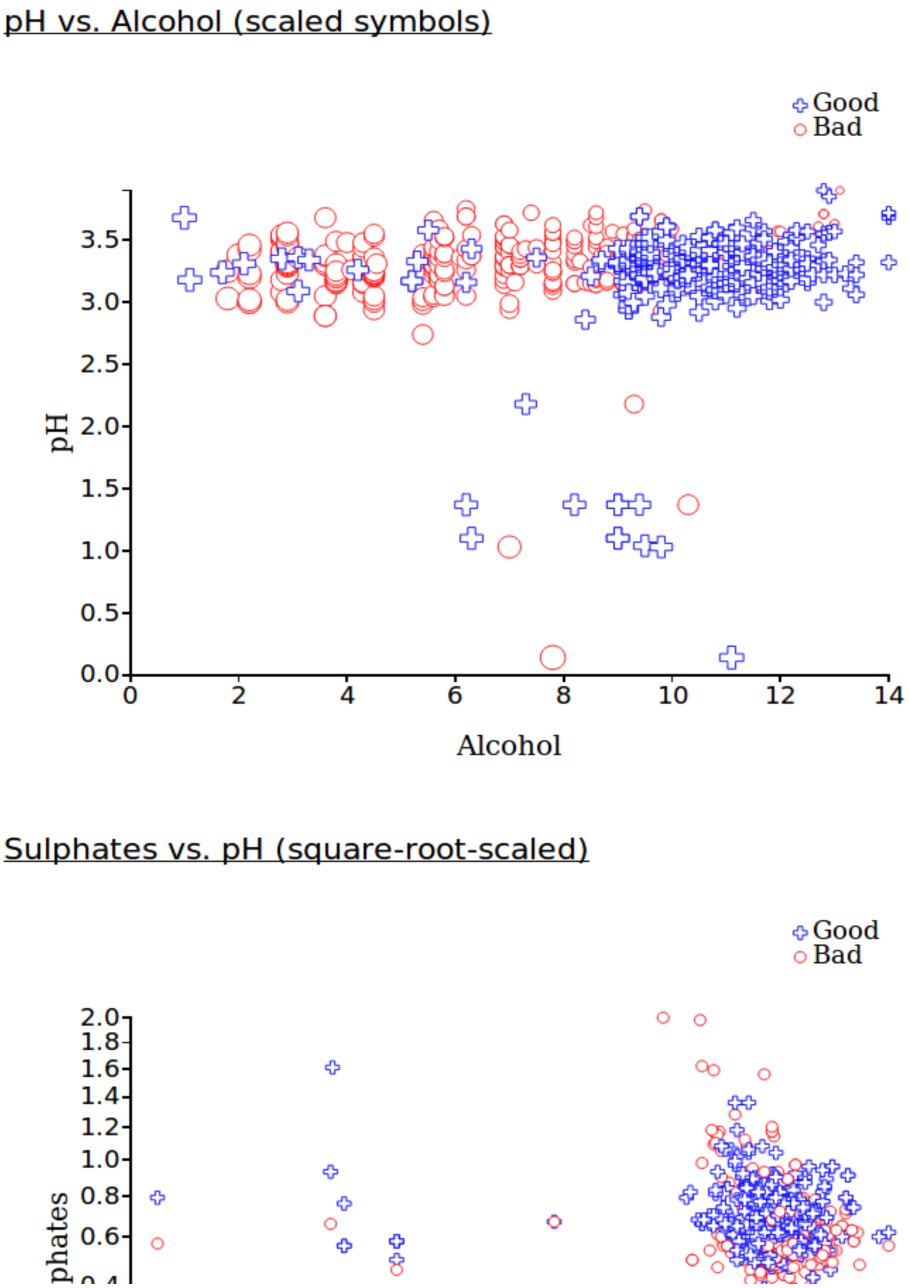

Figure 3: Example for scatter plots, on a single HTML page.

Q3 Deliverables:
The directory structure should be organized as follows:

Q3/

scatterplot.(html / js / css)
explanation.txt
scatter_plots.pdf
wine_quality.csv

- **scatterplot.(html / js / css)** - the html / js / css files created.
- **explanation.txt** - the text file explaining your observations for Q3.C.
- **scatter_plots.pdf** - a PDF document showing the screenshots of the five scatter plots created above (two for Q3.A, one for Q3.B and two for Q3.C). You may print the HTML page as a PDF file, and each PDF page shows one plot. **Hint:** To make it work, one way is to use CSS page break (refer to stackoverflow). Clearly title the plots in the document (see examples in Figure 3), using the following titles:
  - pH vs. Alcohol
  - pH vs. Residual Sugar
  - pH vs. Alcohol (scaled symbols)
  - Sulphates vs. pH (square-root-scaled)
  - Sulphates vs. pH (log-scaled)
- **wine_quality.csv** - the dataset.

Q4 [15 pts] Heatmap and Select Box

Example: 2D Histogram, Select Options

Use the dataset[1] provided in *heatmap.csv* (in the folder Q4) that describes the number of appearances of characters from each six of the houses in HBO's Game of Thrones across episodes and seasons. Visualize the data using D3 heatmaps.

a. **[6 pts]** Create a heatmap of the number of appearances of characters from each house for season 1 of Game of Thrones. Place the episode on the heatmap's horizontal axis and the house on its vertical axis. Number of appearances for each house will be represented by colors in the heatmap. There should be 9 color gradations.

b. **[3 pt]** Add axis labels and a legend to the chart. Place the name of the house ("Baratheon", "Lannister", "Stark", etc.) on the vertical axis and the episode number on the horizontal axis.

c. **[6 pt]** Now create a drop down select box with D3 that is populated with the season numbers (1 to 6) in ascending order. When the user selects a different season in this select box, the heatmap and the legend should both be updated with values corresponding to the selected season. Note the differences in the legends for season 1 and 3 in the images below. While the 9 color gradations in the legend remain the same, the thresholds values are different. The default season when the page loads should be 1 (i.e., the first season).

[1] Source: https://github.com/fredhohman/a-viz-of-ice-and-fire

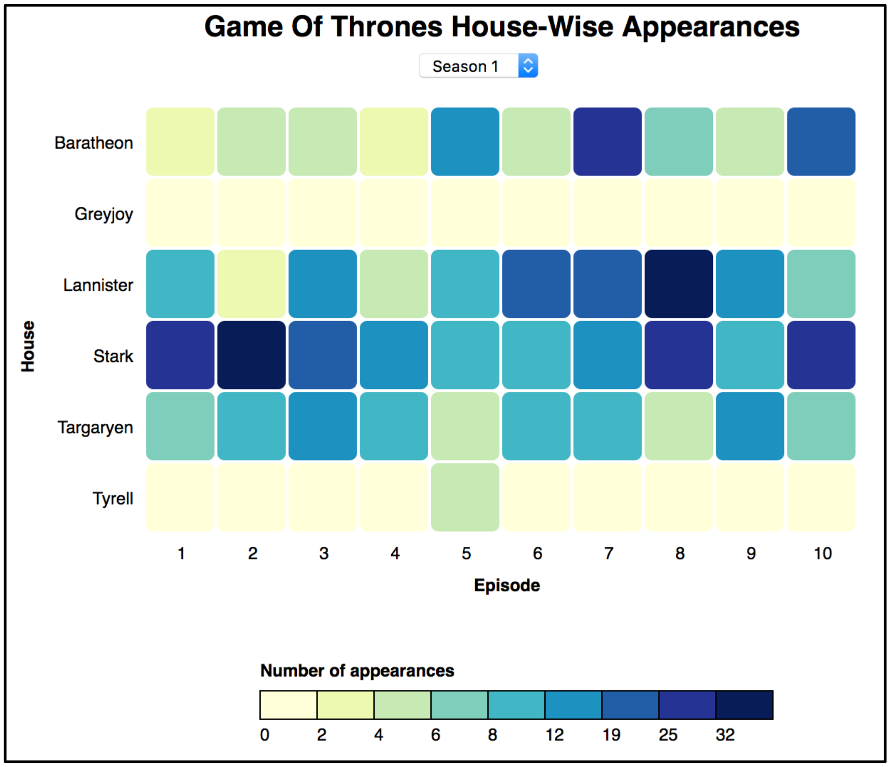

Figure 4a: No of appearances of characters from each house on Season 1 of Game of Thrones

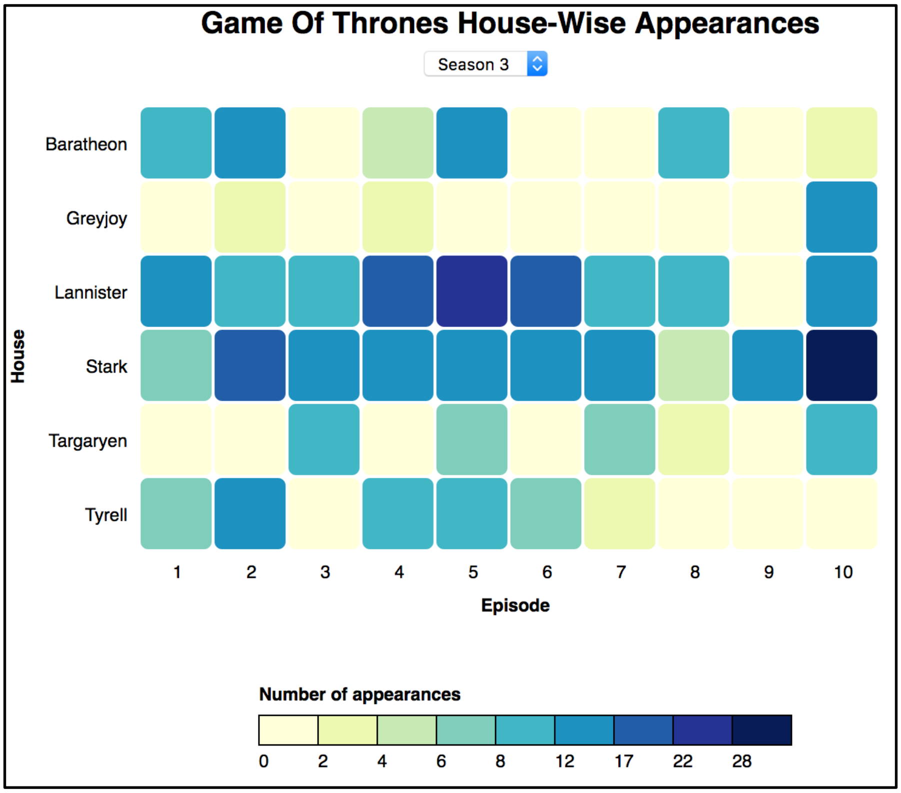

Figure 4b: No of appearances of characters from each house on Season 3 of Game of Thrones

Q4 Deliverables:

The directory structure should look like:

```
Q4/
   heatmap.(html / js /css)
   heatmap.csv
```

- **heatmap.(html / js/ css)** - the html / js / css files created.
- **heatmap.csv** - the dataset

Q5 [20 pts] Visualizing Hierarchical Data with a Dendrogram

Example: Collapsible Tree

In the homework skeleton you will find `sushi.json`, a dataset derived from http://www.kamishima.net/sushi/[2]. The data in this file is organized in a hierarchy, the format of which is shown below. Use D3 to visualize the data as a collapsible dendrogram. Figure 5 shows an example of how your visualization may look like.

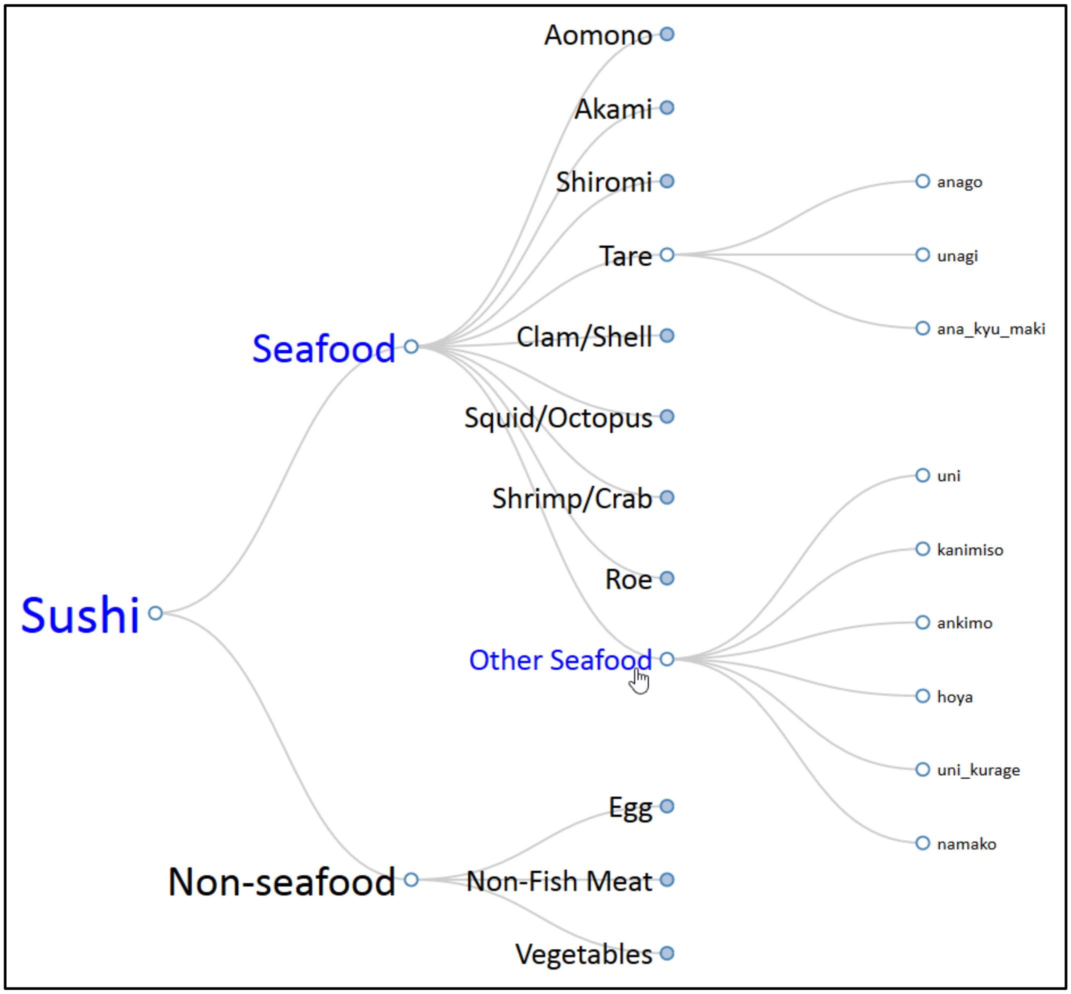

Figure 5. Example dendrogram with cursor hovering over “Other Seafood”

[2] The data derived from http://www.kamishima.net/sushi/ was not originally in json format. D3.stratify() (only available in D3 V4) was used to convert the data into json format for you, so that you can easily visualize it as a dendrogram.

Hierarchy:

- Sushi
    - Seafood
        - Aomono
        - Akami
        - Shiromi
        - Tare
        - Clam/Shell
        - Squid/Octopus
        - Shrimp/Crab
        - Roe
        - Other Seafood
    - Non-seafood
        - Egg
        - Non-Fish Meat
        - Vegetables

a. **[5 pts]** Use D3 to visualize `sushi.json` as a collapsible dendrogram. Collapsible means that when you click on a node, it expands its children if they are not currently displayed and hides them if they are. See the link at the beginning of this question for an example.

b. **[5 pts]** Set the labels' font sizes such that they decrease in size the deeper they are in the tree. For example, "Sushi" is at the highest level in the tree so it should have the largest font size.

c. **[10 pts]** When you hover your mouse cursor over a label or its node, your code should color the label and all its ancestors' labels blue. When you move your cursor away set the colors back to black. For example, in Figure 5, the cursor is hovering over "Other Seafood", so it and its ancestors, "Seafood" and "Sushi", are colored blue. **Note:** You may need to be careful about handling names with special characters, like spaces and dashes.

Q5 Deliverables:
The directory structure should be as follows:

```
Q5/
    index.(html/js/css)
    sushi.json
```

- **index.(html/js/css)** - visualization of dendrogram (part a), including features created in parts b and c
- **sushi.json** - the provided dataset

Q6 [20 pts] Interactive visualization

Use the dataset provided in the dataset.txt file (in the Q6 folder) to create an interactive bar chart. Each line in the file represents a movie franchise and the revenue in millions for each movie in that franchise.

You will have to integrate the data provided in dataset.txt directly into an array variable in the script.

Example: `<script> var data=[<paste data file content here>];</script>`

a. **[5 pts]** Create a **horizontal bar chart** with its vertical axis denoting the movie franchise names and its horizontal axis denoting the total revenue (in millions) of all the movies in the franchise. Each bar should have the total revenue (in millions) labelled inside it. Refer to the example shown in Figure 6a. **Note:** The vertical axis of the chart should use movie franchise names as labels.

b. **[10 pts]** On hovering over a bar, a smaller line chart representing the revenue for each movie (1, 2, 3 etc.) should be displayed in the top right corner. For example, movies from the Fast and the Furious franchise earned a revenue of 206, 236, 363, 363, 629, 789, 1516, and 1237 (in millions) for each installment. On hovering over the bar representing Fast and the Furious, a line chart depicting these 8 values should be displayed. Include axis titles and axis values. See Figure 6b for an example.

c**. [3 pts]** On mouse out, the line chart should no longer be visible.

d. **[2 pts]** On hovering over any horizontal bar representing a movie franchise, the color of the bar should change. You can use any color that is visually distinct from the regular bars. And on mouseout, the color should be reset.

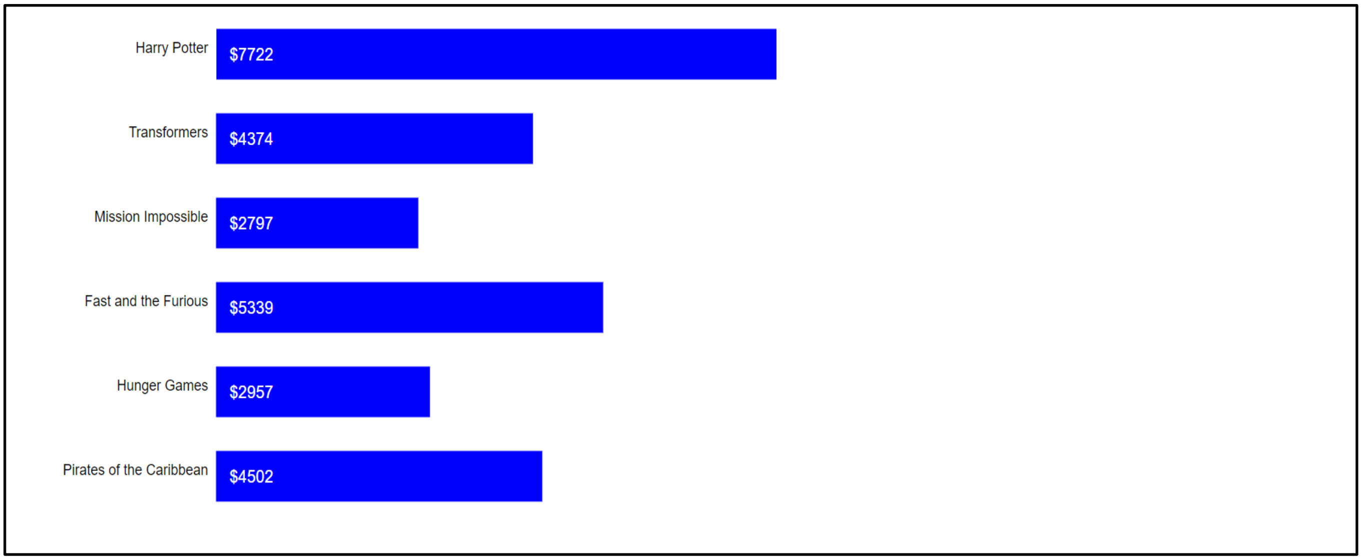

Figure 6a. Bars representing total revenue (in millions) of each movie franchise

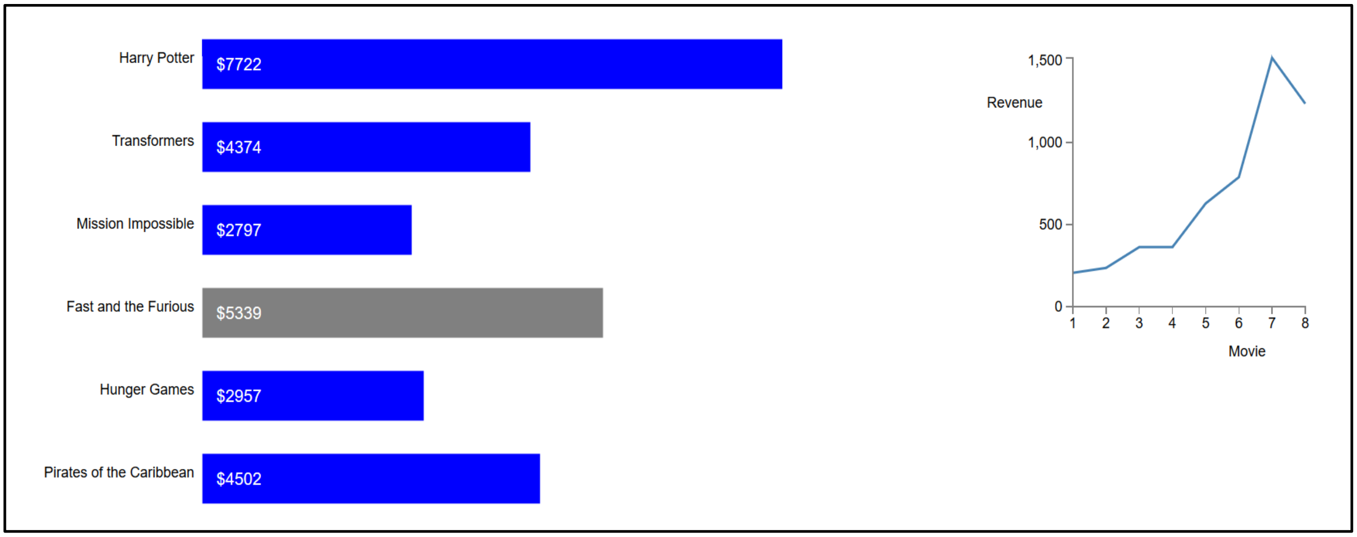

Figure 6b. On hovering over the bar for Fast and the Furious, the color of the bar changes and a smaller line chart representing the revenue in millions of each movie in that franchise is displayed at the top right corner.

Q6 Deliverables:
The directory structure should be as follows:

Q6/
interactive.(html/js/css)

interactive.(html/js/css) - The html, javascript, css to render the visualization in Q6 (dataset.txt is *NOT* required to be included in the final directory structure as the data provided in dataset.txt should have already been integrated into the "data" variable in your code.)

Q7 [20 pts] Choropleth Map of World Population

Example: Unemployment rates

Use the provided dataset in *world_population.tsv*[3], *world_countries.json* and *literacy_rate.tsv*[4] (in the folder Q7) and visualize them as a choropleth map.

- Each record in *literacy_rate.tsv* represents a country's literacy rate and is of the form `<id,Country,Rate>`, where
  - `id` corresponds to the ISO code of the country
  - `Country` is the country's name
  - `Rate` is the total literacy rate of the people (both female and male) living in the country

[3] Data from the World Bank

[4] Data from the UNICEF

- Each record in *world_population.tsv* represents a country's population and is of the form `<id,name,population>`, where
  - `id` corresponds to the ISO code of the country
  - `name` is the country's name
  - `population` is the total population of that country

- The *world_countries.json* file is a [TopoJSON *topology*] containing geometry collections for every *country*.

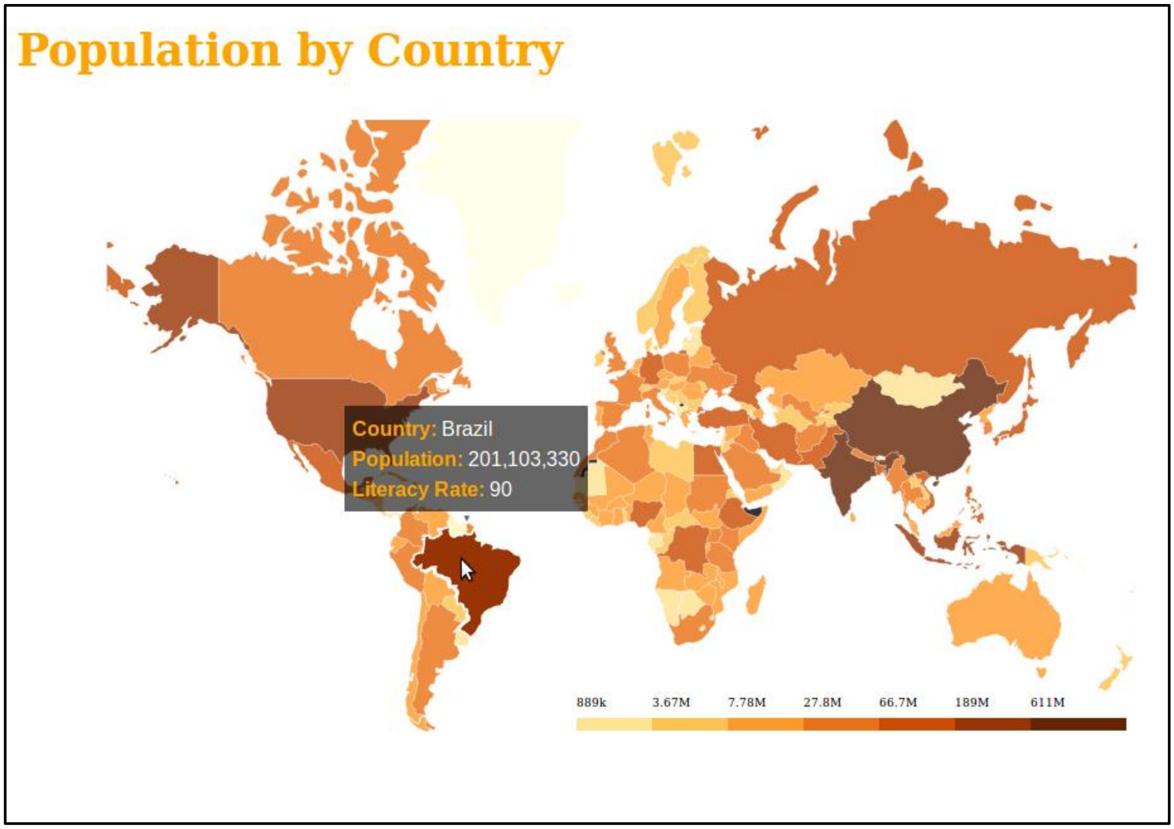

Figure 7. Reference example for Choropleth Maps

a. **[15 pts]** Create a choropleth map using the provided datasets. Color the countries based on their population, i.e., darker colors correspond to higher populations and lighter colors correspond to lower populations. Add a legend showing how colors map to population. Use d3-queue (in the lib folder) to more easily load data from multiple files into a function[5]. Use topojson (present in lib) to draw the choropleth map. Consider using d3.format to shorten the size of the text tick-marks on the legend.

b. **[5 pts]** Add a tooltip using the d3.tip library (in the **lib** folder) that, on hovering over a country, shows the country's name, population and literacy rate. The tooltip should appear on hovering over the state.

[5] d3-queue evaluates a number of asynchronous tasks concurrently -- in this question, each task would be loading one data file. When all tasks have finished, d3-queue passes the results to a user-defined callback function.

On mouseout, the tooltip should disappear.
Note: You must create the tooltip by only using **d3.tip.v0.6.3.js** present in the **lib** folder.

Q7 Deliverables:
The directory structure should be organized as follows:

```
Q7/
   q7.(html/js/css)
   literacy_rate.tsv
   world_population.tsv
   world_countries.json
```

- **q7.(html /js /css**)- The html/js/css file to render the visualization.
- **literacy_rate.tsv** and **world_population.tsv** - The datasets used.
- **world_countries.json** - Dataset needed to draw the map.

Important Instructions on Folder structure

The directory structure must be as follows. The files that should be included in each question's folder (e.g., Q1 for question 1) have been clearly specified at the end of each question's problem description above.

```
HW2-LastName-FirstName/
            |--- lib/
                   |----    d3.v3.min.js
                   |----    d3.tip.v0.6.3.js
                   |----    d3-queue.v3.min.js
                   |----    topojson.v1.min.js
            |--- Q1/
                   |----   ...
            |--- Q2/
                   |----   ...
            |--- Q3/
                   |----   ...
            |--- Q4/
                   |----   ...
            |--- Q5/
                   |----    ...
            |--- Q6/
                   |----    ...
            |--- Q7/
                   |----    ...
```