CSE 6242 / CX 4242: Data and Visual Analytics | Georgia Tech | Fall 2016

Homework 2 : D3 Graphs and Visualization

Due: Monday, October 10, 2016, 11:55 PM EST

Prepared by Nilaksh Das, Pradeep Vairamani, Vishakha Singh, Yanwei Zhang,
Bhanu Verma, Meghna Natraj, 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.

- 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 .rar, etc.)
- 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 (e.g., how cross-validation works) and strategies (e.g., 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 up, 5% deduction for every 24 hours of delay. (e.g., 5 points for a 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 120 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 120 pts, that student will receive (120 / 100) * 15 = 18 pts towards the final course grade. To receive the full 15% score, students in the CSE6242 sections will need to complete all 120 points.

Prerequisites

Download this zip file that contains datasets and files in the correct folder structure you will need for this assignment.

For this homework, you will work with version 3 of D3, provided to you in the **lib** folder.

You may need to setup your own 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). See here for details.

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.

Include all the files related to d3*.js in the **lib** folder, so html files (e.g., those in folders Q2, Q3, etc.) can reference them using relative paths, e.g., "../lib/<filename>"
This also means that you should run the local server in your root homework folder.

Q1 [10 pts] Designing a good table. Visualizing data with Tableau.

Imagine you are a data scientist working with the United Nations High Commissioner for Refugees (UNHCR) and need to perform the following tasks to aid UNHCR's understanding of persons of concern.

a. [5 pts] **Table.** Create a table to display the details of the refugees (Total Population) in the year 2005 from the data provided in *unhcr_persons_of_concern.csv*. You can use any tool (e.g., Excel, HTML) to create the table. Keep suggestions from class in mind when designing your table (see lectures slides for what to and what not to do, but you are not limited to the techniques described). Describe your reason for choosing the techniques you use in **explanation.txt** in no more than 50 words**.**

b. [5 pts] **Tableau:** Visualize the demographic attributes (age, sex, country of origin, asylum seeking country) in the file *unhcr_popstats_demographics.csv* (in the folder Q1) for any given year in one chart. Tableau is a popular InfoViz tool and the company has provided us with student licenses. Go to this link 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.

Provide a rationale for your design choices in this step in the file **explanation.txt** in no more than 50 words.

Q1 Deliverables:
The directory structure should be as follows:

Q1/
table.xxx
chart.yyy
explanation.txt
unhcr_persons_of_concern.csv
unhcr_popstats_demographics.csv

- **table.xxx** - An image/screenshot of the table in Q1.a (png or pdf format).
- **chart.yyy** - An image of the chart in Q1.b (png or pdf format). The image should be clear and of high-quality.
- **explanation.txt** - Your explanations for parts Q1.a and Q1.b in this file.
- **unhcr_persons_of_concern.csv** and **unhcr_popstats_demographics.csv** - the datasets

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 folder Q2).

a) [3 pts] **Adding node labels**: Modify the graph.html to show labels to the right of each node in the graph. If a node is dragged, its label must also move with the node. (You are welcome to split graph.html into graph.html, graph.js and graph.css.)

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 >= 1.5 : assign Blue color to the link
If the value of the edge is < 1.5 : assign Green color to the link.

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

i) Adjust the radius of each node in the graph based on the degree of the node.

ii) In **explanation.txt**, using no more than 40 words, discuss which metric (possible metrics: scaling the radii linearly, scale the radii by the square root of the degree, etc.) you have used and explain why you think it is a good choice.

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

i) 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).

ii) Mark pinned nodes so that they are visually distinguishable from unpinned nodes, e.g., pinned nodes shown with a different color, or border thickness, or visually annotated with a "star" (*), etc.

iii) 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

explanation.txt

Q3 [15 pts] Visualizing scatter plots

Use the dataset[1] provided in the file *iris.tsv* (in the folder Q3) to create a scatterplot.

Refer to the tutorial for scatter plot here.

Features/ Attributes in the dataset:

1. Sepal length in cm
2. Sepal width in cm
3. Petal length in cm
4. Petal width in cm
5. Class: Iris Setosa, Iris Versicolor, Iris Virginica

i. [8 pts] **Creating scatter plots**:

1. [6 pts] Create two scatter plots, one for each feature combination specified below. In the scatter plots, visualize the different classes using different symbols (circle for setosa, square for versicolor and triangle for virginica) and add a legend showing how symbols map to the classes
    - Features 1 and 2
    - Features 3 and 4
2. [2 pts] In **explanation.txt**, using no more than 40 words, discuss which plot is better at separating the classes and why.

Scatter plots should be placed one after the other in an html page as shown in the reference below. Please note that your design need not be identical to the given reference.

[1] Source: https://archive.ics.uci.edu/ml/datasets/Iris

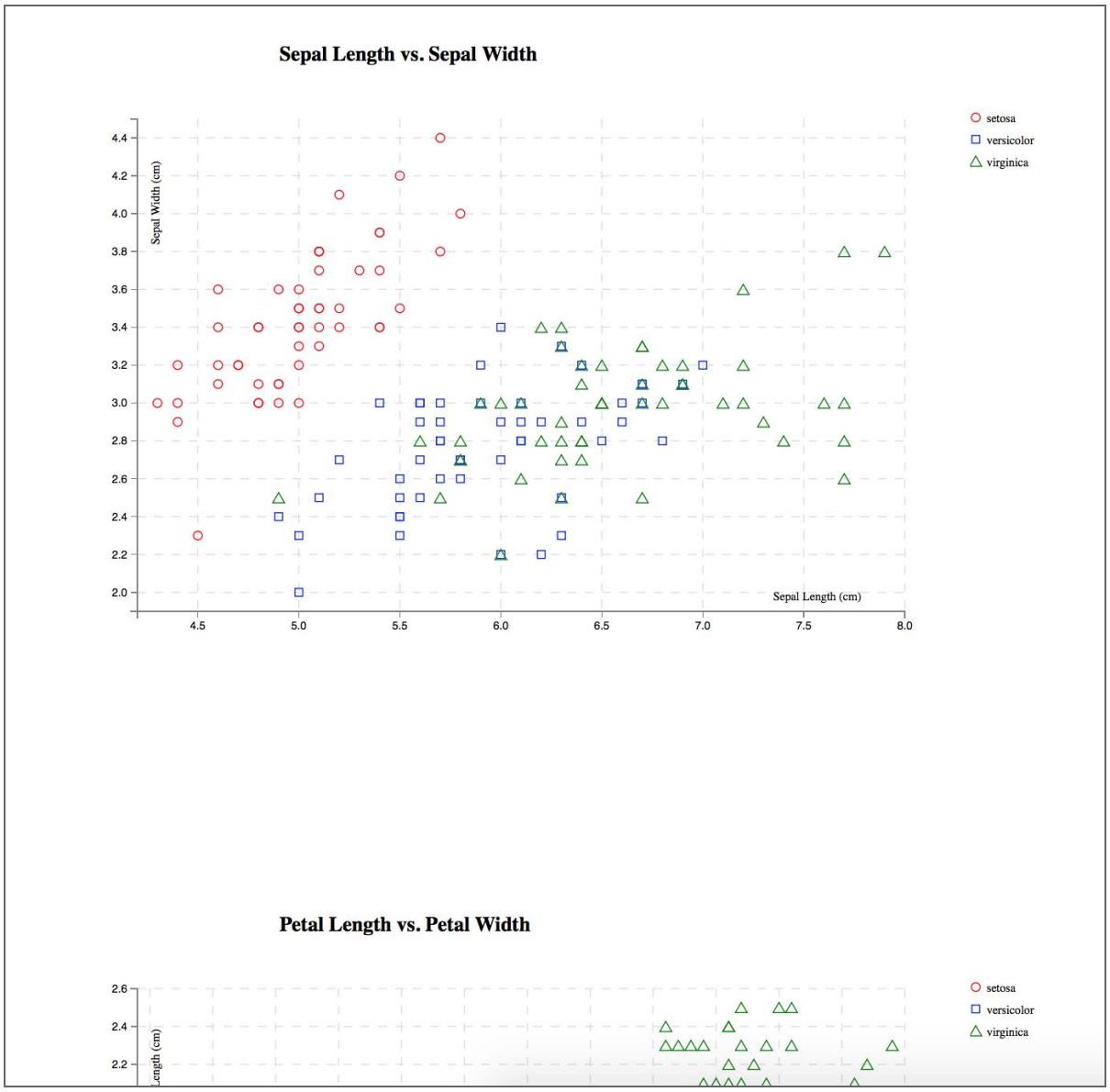

Figure 3 : Reference example for scatter plots

Based on the scatter plot created for features 1 and 2 (Sepal Length vs Sepal Width), create new plots for the following questions:

ii. [3 pt] **Scaling symbol sizes.** Set the size of each symbol in the plot to be proportional to the squared value of the the length parameter. Create a new plot for this part.

iii. [4 pts] **Axis Scales in D3.** Create two plots for this part to try out two axis scales in D3, one for using the square root scale (applied to both axes) and another for using the log scale (also applied to both axes). Explain in no more than 40 words, in **explanation.txt**, when we may want to use log scales in charts (e.g., in scatter plots).

Q3 Deliverables:

The directory structure should be organized as follows:

Q3/

scatterplot.(html / js / css)

explanation.txt
scatter_plots.yyy
iris.tsv

- **scatterplot.(html / js / css)** - the html/js/css files created.
- **explanation.txt** - the text file explaining your approaches for Q3.i.2 and Q3.iii.
- **scatter_plots.yyy** - a screenshot (png or pdf format) showing the five scatter plots created above (two for Q3.i.1, one for Q3.ii and two for Q3.iii).
- **iris.tsv** - the dataset

Q4 [15 pts] Visualizing heat map

Use the dataset[2] provided in *hourly_heatmap.json* (in the folder Q4) that describes glucose readings over time, and visualize it using D3 heatmaps. To get started, refer to the heatmap example here.

i. [7 pts] Plot the glucose readings against the time of the day (Hint: Use the glucose readings as a "**z"** parameter in the given example)
ii. [6 pts] Now use the file *day_heatmap.json* (in the folder Q4) to plot the glucose readings against the day of the week on the heatmap. Use the day names instead of numbers as the tick labels on the axis, e.g., day 1 being Monday.
iii. [2 pt] A pattern should emerge from the visualizations. Explain the pattern and why it occurs, using no more than 40 words in **explanation.txt**.

Please note that there will be two heat maps, one for part i and the other for part ii. Place them one after the other on an html page (the one for part i goes first).

Q4 Deliverables:
The directory structure should look like (remember to include the d3 library):

Q4/
heatmap.(html / js /css)
heatmap.yyy
explanation.txt
day_heatmap.json
hourly_heatmap.json

- **heatmap.(html / js/ css)** - the html / js / css files created.
- **heatmap.yyy** - a screenshot (png or pdf format) of the plots created in Q4.ii
- **explanation.txt** - the text file explaining your answer for Q4.iii Keep it succinct (each answer should be no more than 40 words).
- **day_heatmap.json** and **hourly_heatmap.json** - the datasets

[2] Source: https://github.com/jebeck/iPancreas-archive

Q5 [25 pts] Sankey Chart

Formula One racing is a championship sport in which race drivers represent teams to compete for points over several races (also called Grand Prix) in a season. The team with the most points at the end of a season wins the prestigious Formula One World Constructors' Championship award. You will visualize the flow of points for the races held in this season up to September 2016[3]. The drivers win points according to their final standing in each race, which finally get added to their respective team's total.

The implementation of certain parts in this question may be quite challenging.

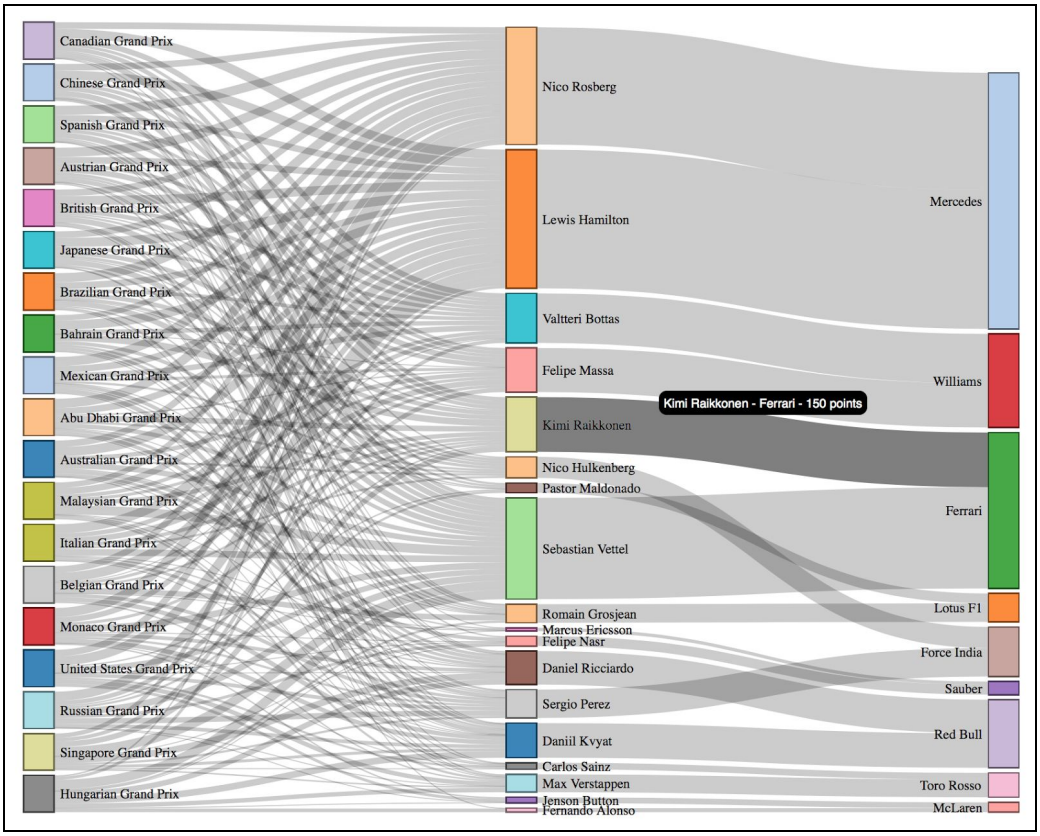

Figure 5 : Example Sankey Chart visualizing the flow of points for the 2015 season

a. [15 pts] Create a *Sankey Chart* using the datasets provided (*races.csv* and *teams.csv*) in the Q5 folder. The chart should visualize the flow of points in the order:

```
race → driver → team
```

You may refer to this example to create the chart (sankey.js is provided in the lib folder). You can keep the blocks' vertical positions static. Your chart should look like the example Sankey Chart for the 2015 season as shown in Figure 5.

Hint : For this part, you will have to read in the csv files and combine the data into a format that can be passed to the sankey library. To accomplish this, you may find the following javascript functions useful: d3.nest(), array.filter(), array.map()

[3] Source: http://ergast.com/mrd/

b. [5 pts] Use the d3-tip library to add tooltips as shown in Figure 5 (you can make your own visual style choices using css properties).

c. [5 pts] From the visualization you have created, determine the following:
   i. Which team has the best current standing? [1 pt]
   ii. Which driver has the most points currently? [1 pt]
   iii. Which driver won the Monaco Grand Prix? [1 pt]
   iv. Which two drivers switched their teams mid-season? [2 pts]

   Put your answers in **observations.txt**.

Q5 Deliverables:
The directory structure should be as follows:

Q5/
races.csv
teams.csv
viz.(html/js/css)
observations.txt

- **viz.(html/js/css)** - The html, javascript, css to render the visualization in Q5.a.
- **observations.txt** - Write your answer for Q5.c in this file.
- **races.csv** and **teams.csv** - the datasets

Q6 [20 pts] Interactive visualization

Mr. Fluke runs a small company named FooBar. His company manufactures eight products around the year. He wants you to create an interactive visualization report using D3 so that he can see the total revenue generated per product type and the revenue breakdown across product types for the four quarters in 2015. Use the dataset provided in the Q6 folder. Integrate the dataset provided in dataset.txt directly in an array variable in the script. Example: `<script> var data=[<include data here>];</script>`

a. **[5 pts]** Create a **horizontal bar chart** with its vertical axis denoting the product names and its horizontal axis denoting the total revenue. Each bar should have the total revenue amount in dollars labelled inside it. See Figure 6 for an example.

b. **[5 pts]** Create a **legend** with three columns.
Column 1: quarter labels: Q1, Q2, Q3, Q4
Column 2: initialized with each quarter's total revenue (e.g., Q1's value is initialized as the sum of all products' revenues in Q1)
Column 3: presents the percentage share of each value in Column 2

c**. [10 pts]** While hovering over any bar, the second and third columns in the legend should **update** to

show the revenue generated (in value and percentage share, respectively) for each quarter of the selected product. For example, when hovering over Product C's bar, the second and third columns in the legend should update to show Product C's revenues in the four quarters and those revenues' percentage shares. See Figure 6 for an example.

Note:

1. The vertical axis of the chart should use product names as labels.
2. On hovering over any horizontal bar, the color of the bar should change. You can use *any* color that is visually distinct from the regular bars.
3. The legend should reset to the initial values on mouseout (i.e., when the mouse leaves a bar).

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).

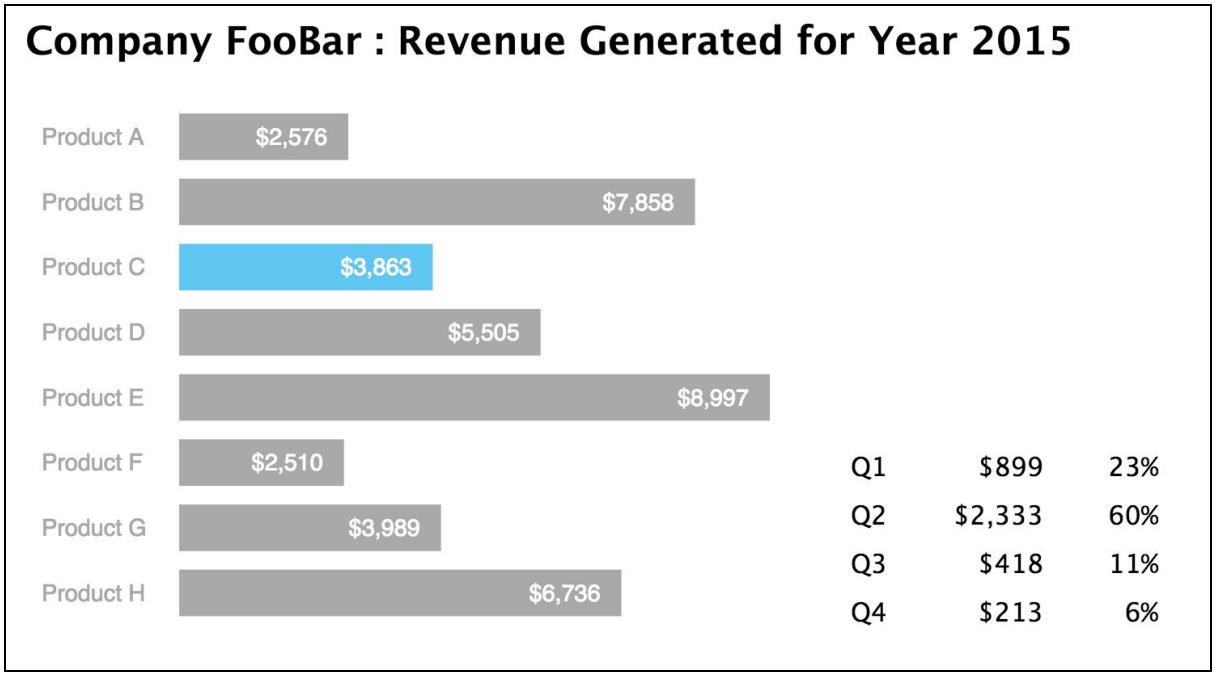

Figure 6

Q7 [20 pts] Visualizing college scorecard data

This is a free-form question. We want you to apply the D3 knowledge that you have gained to assist decision making for a real-world problem: help students make college decisions.

Using D3, construct a visualization using the college scorecard dataset (**located in the Q7 folder**) which contains statistics about colleges (e.g., affordability, value).

Create one large visualization or multiple small ones using the entire dataset or a subset of it. If you want, you may also use the Bootstrap library, which is a popular framework used in frontend development, to organize your dashboard -- we recommend Bootstrap because many student teams in previous semesters had good experiences using it for their projects. Place the Bootstrap library files in the **lib/bootstrap/** folder. The visualization does not need to support any interactions.

You may also refer to the information on this website to augment your data or for more ideas: http://catalog.data.gov/dataset/college-scorecard

- Points will be awarded for usability, functionality, and creativity.
- Summarize your main ideas behind the visualization in **explanation.txt** in no more than 50 words**.**

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

Q7/
q7.(html /jss /css)
*.json
explanation.txt

- **q7.(html /js /css**)- The html, javascript, css files to render the visualization.
- ***.json** - Include all the json file(s) used as data sources.
- **explanation.txt** - Summarize your idea behind the visualization in no more than 50 words.

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
              |----    sankey.js
        |--- Q1/
              |----    ...
        |--- Q2/
              |----    ...
        |--- Q3/
              |----    ...
        |--- Q4/
              |----    ...
        |--- Q5/
              |----    ...
        |--- Q6/
              |----    ...
        |--- Q7/
              |----    ...
```