CSE 6242/CX 4242: Data and Visual Analytics | Georgia Tech | Fall 2017

Homework 2 : D3 Graphs and Visualization

Due: Wednesday, October 11, 2017, 11:55 PM EST

Prepared by Kiran Sudhir, Varun Bezzam, Yuyu Zhang, Akanksha Bindal,
Vishal Bhatnagar, Vivek Iyer, 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 "HW1-{YOUR_LAST_NAME}-{YOUR_FIRST_NAME}.zip", containing all the deliverables including source code/scripts, data files, and readme. Example: 'HW1-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 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.

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 55.0.2 (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>
```

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. Visualizing data with Tableau.

Imagine you are a data scientist working with United Nations High Commissioner for Refugees (UNHCR) and the Uniform Crime Reporting division of Federal Bureau of Investigation (FBI). Perform the first subtask to aid UNHCR's understanding of persons of concern and the second subtask to aid FBI in analysing changing crime rates.

a. **[5 pts] Good table design.** Create a table to display the details of the refugees (Total

Population) in the year 2012 from the data[1] 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, 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.txt** in no more than 50 words**.**

b. **[5 pts] Tableau:** Visualize the change in different crime rates (e.g., Burglary rate, Property Crime Rate, etc) in the dataset[2] *crime_rates_FBI.csv* (in Q1 folder) over the given years, using a line chart.
   - Your chart should visualize at least 3 crime rates over time; you are welcome to choose which specific crime rates to visualize.
   - *Scale* the thickness of each trend line based on Population count.
   - **Save the chart as timeseries.(png/pdf).**

   Tableau is a popular information visualization tool and the company 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)
   timeseries.(png / pdf)
   explanation.txt
   unhcr_persons_of_concern.csv
   crime_rates_FBI.csv
```

- **table.(png / pdf)** - An image/screenshot of the table in Q1.a (png or pdf format **only**).
- **timeseries.(png / pdf)** - An image of the chart in Q1.b (png or pdf format **only**, Tableau workbooks will not be graded!). The image should be clear and of high-quality.
- **explanation.txt** - Your explanation for part Q1.a in this file.
- **unhcr_persons_of_concern.csv** and **crime_rates_FBI.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 Q2 folder). **Note:** You are welcome to split graph.html into graph.html, graph.css, and graph.js.

[1] Source: http://popstats.unhcr.org/en/overview
[2] Source: https://ucr.fbi.gov/

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 Green color to the link.
> If the value of the edge is >= 3.0 and <= 4.0 : assign Red color to the link.
> If the value of the edge is > 4.0 : assign Blue 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
    explanation.txt
    graph.js, graph.css (if not included in graph.html)
```

- **graph.html** - the html file created.
- **explanation.txt** - the text file explaining your design choices for Q2.
- **graph.(js / css)** - the js / css files if not included in graph.html

Q3 [15 pts] Scatter plots

Use the dataset[3] provided in the file *diabetes.csv* (in the folder Q3) to create a scatterplot.

Refer to the tutorial for scatter plot here.

[3] Source: http://archive.ics.uci.edu/ml/datasets/Pima+Indians+Diabetes

Attributes in the dataset:

Feature 1: Number of times pregnant
Feature 2: Plasma glucose concentration at the 2nd hour in an oral glucose tolerance test
Feature 3: Diastolic blood pressure (mm Hg)
Feature 4: Triceps skin fold thickness (mm)
Feature 5: 2-hour serum insulin (mu U/ml)
Feature 6: Body mass index (weight in kg/(height in m)^2)
Feature 7: Diabetes pedigree function
Feature 8: Age (years)
Class: 0 or 1 (class value 1 means "tested positive for diabetes")

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

   1. **[6 pts] Create two scatter plots**, one for each feature combination specified below. In the scatter plots, visualize "negative" class instances as blue circles, and "positive" instances as red triangles. Add a legend showing how symbols map to the classes.
      - Features 2 (plasma glucose) vs. Feature 5 (insulin)
      - Features 6 (BMI) vs. Feature 3 (blood pressure)

   2. **[2 pts]** In **explanation.txt**, use no more than 50 words to discuss which feature combination is better at separating the classes and why.

Your scatter plots should be placed one after the other **on a single HTML page**, similar to the example image below (Figure 3). Note that your design needs NOT be identical to the example.

Based on the scatter plot created for Feature 2 and Feature 5 (Plasma Glucose vs. Insulin), create new plots for the following questions:

B. **[3 pts] Scaling symbol sizes.** Set the size of each symbol in the plot to be proportional to the product of plasma glucose and insulin values. **Create a new scatter plot for this part** (append to the HTML page). Set the scaling coefficient properly to make the scatter plot legible.

C. **[4 pts] Axis scales in D3.** Create two plots for this part (append to the HTML page) to try out two axis scales in D3: the first one uses the square root scale for its y-axis (only), and the second one uses the log scale for its y-axis (only). Note that the x-axes (Plasma Glucose) should be kept in linear scale, and only the y-axes (Insulin) 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.

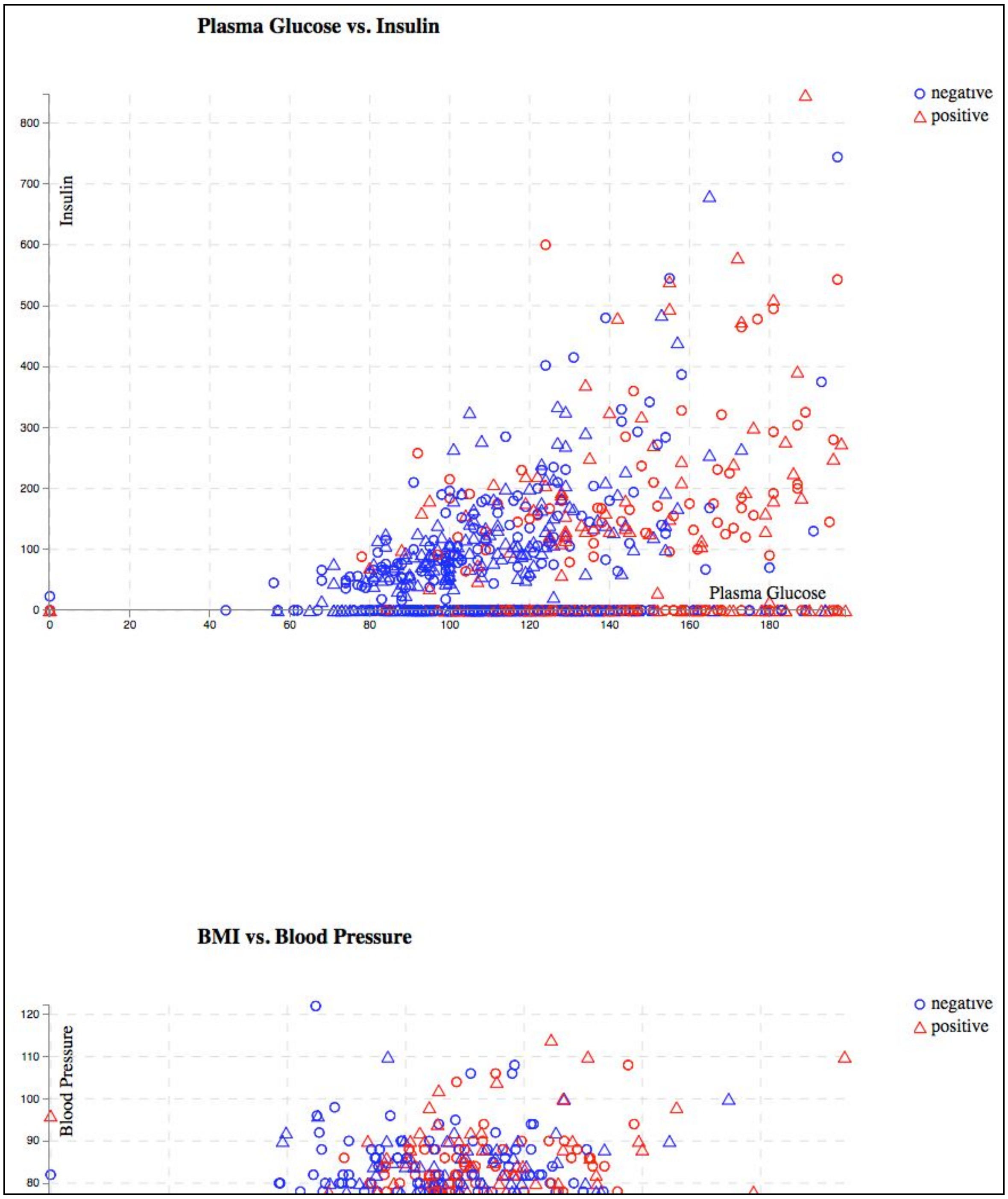

Figure 3: Example for scatter plots, on a single HTML page.

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

Q3/

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

- **scatterplot.(html / js / css) -** the html / js / css files created.
- **explanation.txt -** the text file explaining your observations for Q3.A.2 and Q3.C.
- **scatter_plots.pdf** - a PDF document showing the screenshots of the five scatter plots created above (two for Q3.A.1, 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:
    - Plasma Glucose vs. Insulin
    - BMI vs. Blood Pressure
    - Plasma Glucose vs. Insulin, scaled symbols
    - Plasma Glucose vs. Insulin (square-root-scaled)
    - Plasma Glucose vs. Insulin (log-scaled)
- **diabetes.csv** - the dataset.

Q4 [15 pts] Heatmap and Select Box

Example: 2D Histogram, Select Options

Use the dataset[4] provided in *heatmap.csv* (in the folder Q4) that describes the number of appearances of characters from each house 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).

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

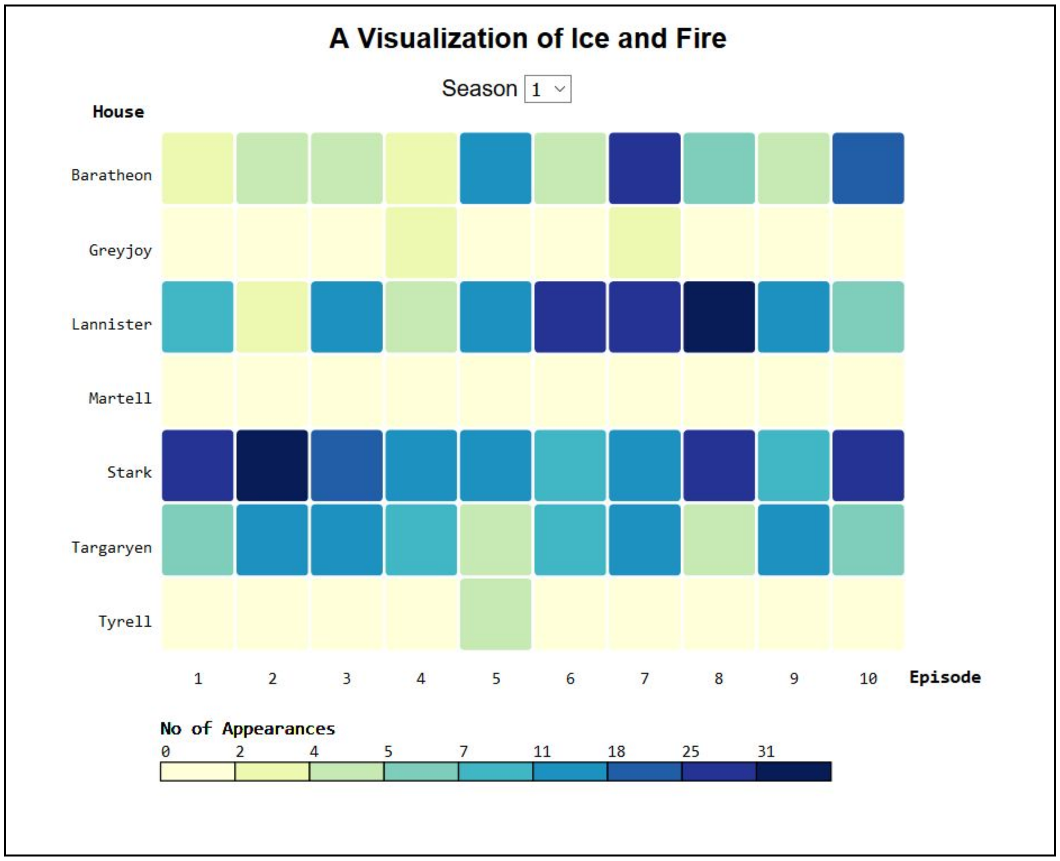

Figure 4a: No of appearances of characters from each house on Season 1 of Game of Thrones

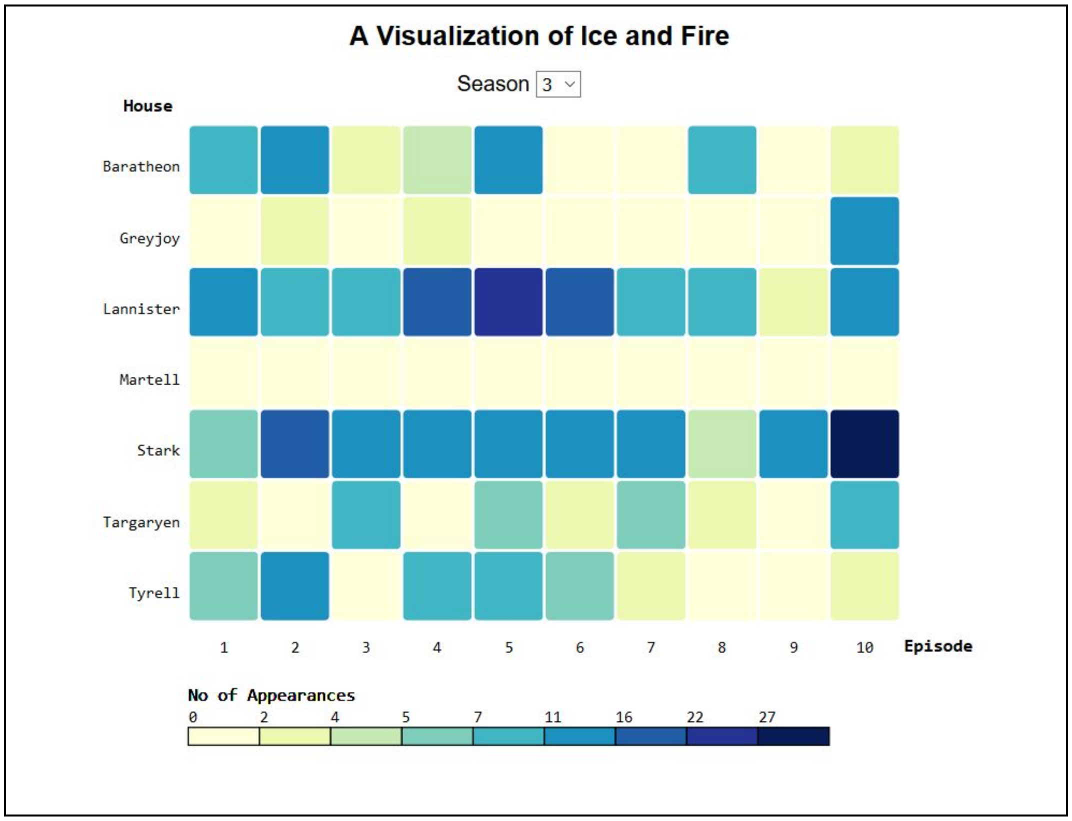

Figure 4b: No of appearances of characters from each house on Season 3 of Game of Thrones

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

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

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

Q5 [25 pts] Sankey Chart

Example: [Sankey diagram from formatted JSON]()

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 2016[5]. The drivers win points according to their final standing in each race, which finally get added to their respective team's total.
Note: The implementation of certain parts in this question may be quite challenging.

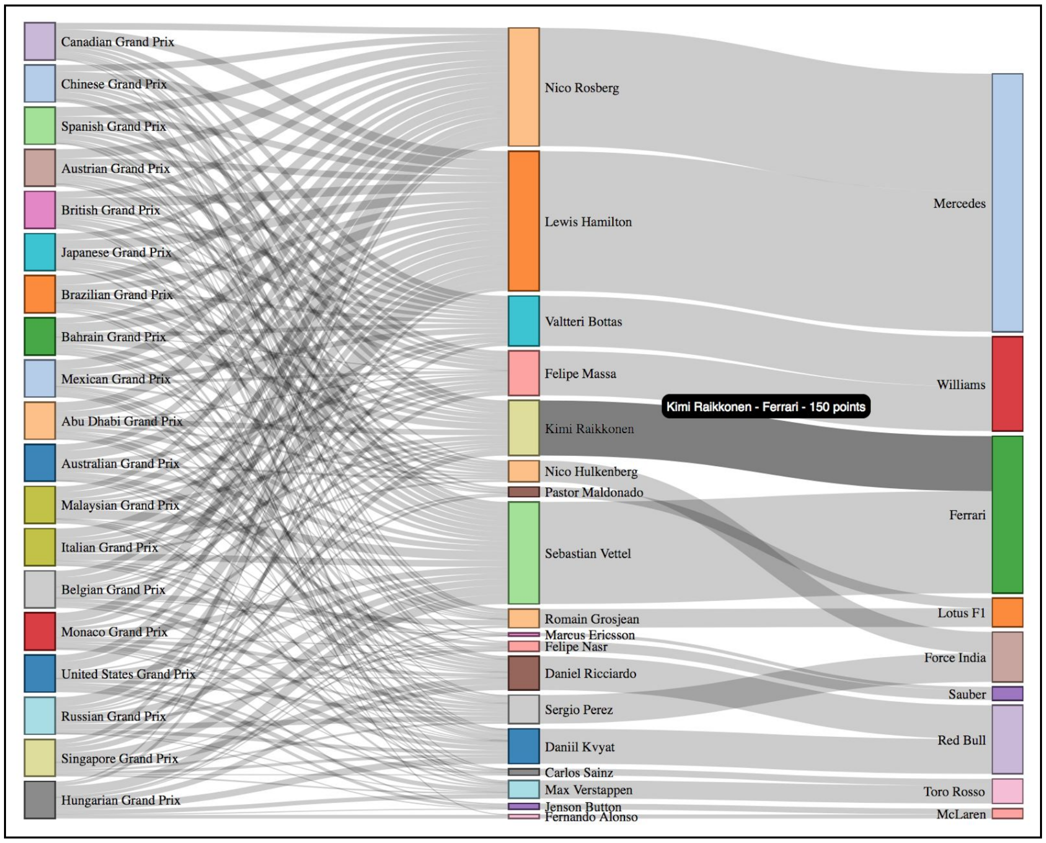

Figure 5. Example Sankey Chart visualizing the flow of points for the 2015 season

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

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

```
race → driver → team
```

You must use the **sankey.js** provided in the **lib** folder. You can keep the blocks' vertical positions static. Your chart should look similar to the example Sankey Chart for the 2015 season as shown in the above image.
Note: 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()

b. **[6 pts]** Use the d3-tip library to add tooltips as shown in the above image. You are welcome to make your own visual style choices using css properties.
Note: You must create the tooltip by only using **d3.tip.v0.6.3.js** present in the **lib** folder.

c. **[4 pts]** From the visualization you have created, determine the following:
   1. [1 pt] Which driver won the Grand Prix 2016?
   2. [1 pt] Which team won the Grand Prix 2016?
   3. [1 pt] Which driver won the Spanish Grand Prix?
   4. [1 pt] Which team has the highest number of players?

   Put your answers in **observations.txt**. Modify the template provided to you (in Q5 folder) by replacing team_name/driver_name with your answer

Sample **observations.txt**
1.driver_name 2.team_name 3.driver_name 4.team_name

Q5 Deliverables:
The directory structure should be as follows:

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

- **races.csv** and **teams.csv** - the data sets (unmodified)
- **viz.(html/js/css)** - The html, javascript, css to render the visualization in Q5.a and b.
- **observations.txt** - Your answer for Q5.c.

Q6 [20 pts] Interactive visualization

Use the dataset[6] provided in the data.txt file (in the Q6 folder) to create an interactive bar chart. Each line in the file represents an English football club, and its value in millions over the past five years. You will have to integrate the data provided in dataset.txt directly in 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 club names and its horizontal axis denoting the total values (in millions) over the past 5 years. Each bar should have the total value (in millions) labelled inside it. Refer to the example shown in Figure 6a.
Note: The vertical axis of the chart should use club names as labels.

b. **[10 pts]** On hovering over a bar, a smaller line chart representing the value of that club for each year (2013-2017) should be displayed in the top right corner. For example, Liverpool has a value of $651M, $704M, $982M, $1548M, $1492M for the years 2013, 2014, 2015, 2016 and 2017 respectively. On hovering over the bar representing Liverpool, a line chart depicting these 5 values is displayed. 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 club, the color of the bar should change. You can use any color that is visually distinct from the regular bars. On mouseout, the color should be reset.

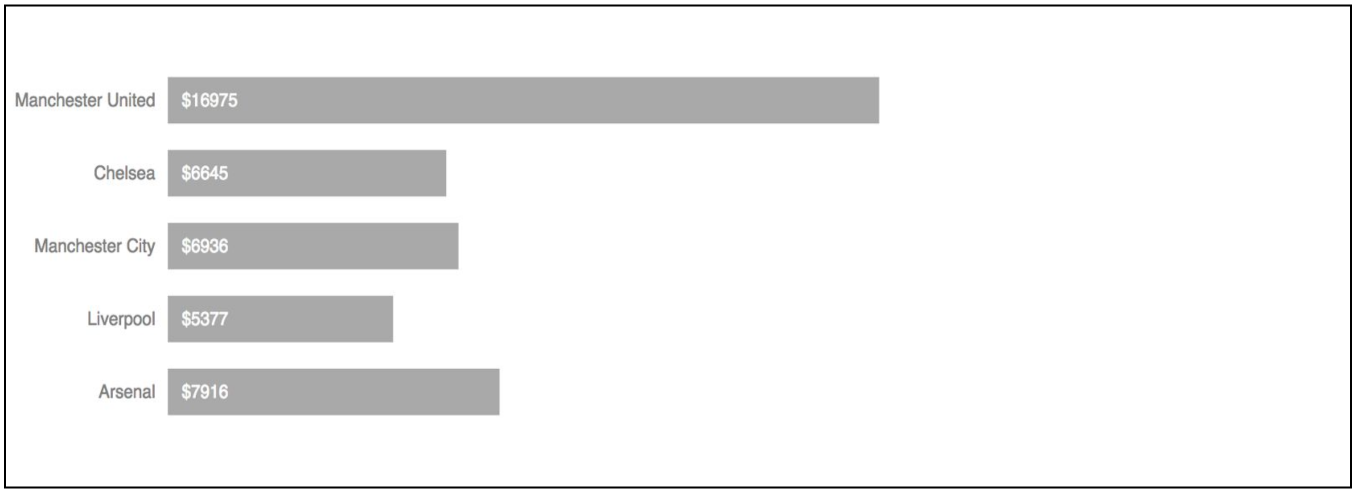

Figure 6a. Bars representing total value (in millions) of each club

[6] Source: Forbes yearly soccer rankings

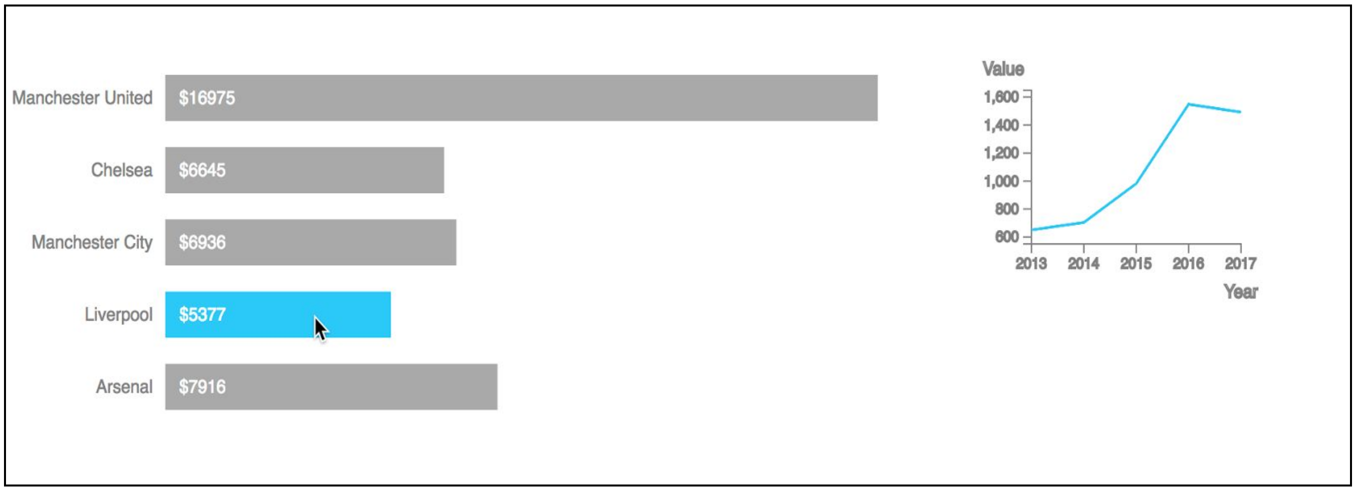

Figure 6b. On hovering over the bar for Liverpool, a smaller line chart representing its value in millions over the past 5 years is displayed at the top right corner.

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 County Data

Example: Unemployment rates

Use the provided dataset[7] in *median_ages.csv*, *us.json* and *median_earnings.json* (in the folder Q7) and visualize them as a choropleth map.

- Each record in *median_ages.csv* represents a county and is of the form `<id,name,median_age>`, where
  - `id` corresponds to the state the county is in
  - `name` is the county name
  - `median_age` is the median age of the people living in the county
- The *median_earnings.json* file contains a list of JSON objects, each having two fields: an `id` field corresponding to a state in the United States, and a `median_earnings` field corresponding to the median earnings of people in that state after 10 years.
- The *us.json* file is a TopoJSON *topology* containing three geometry collections: *counties*, *states*, and *nation*.

[7] Source: Derived from Data USA.

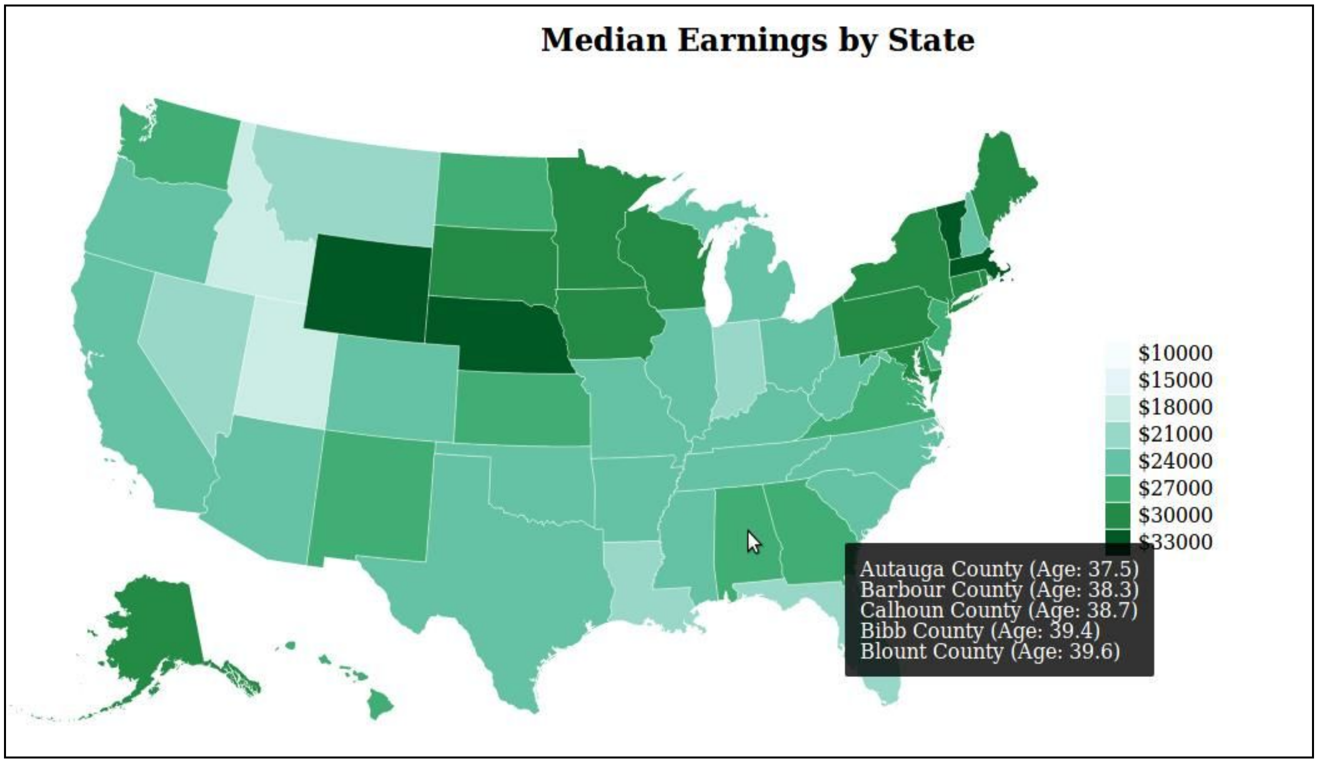
Median Earnings by State
$10000
$15000
$18000
$21000
$24000
$27000
$30000
$33000
Autauga County (Age: 37.5)
Barbour County (Age: 38.3)
Calhoun County (Age: 38.7)
Bibb County (Age: 39.4)
Blount County (Age: 39.6)

Figure 7. Reference example for Choropleth Maps

a. **[15 pts]** Create a choropleth map using the provided datasets. The color of each state should correspond to the median earnings in that state, i.e., darker colors correspond to higher median earnings in that state and lighter colors correspond to lower median earnings in that state. Add a legend showing how colors map to median earnings. Use d3-queue (in the lib folder) to easily load data from multiple files into a function[8]. Use topojson (present in lib) to draw the choropleth map.

b. **[5 pts]** Add a tooltip using the d3.tip library (in the **lib** folder) that, on hovering over a state, shows the **5** counties in that state with the lowest median ages in ascending order, along with those ages. If a state has fewer than 5 counties, show all counties available, along with their median ages. The tooltip should appear on hovering over the state. 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)
   median_ages.csv
   median_earnings.json
   us.json
```

- **q7.(html /js /css)**- The html/js/css file to render the visualization.
- **median_ages.csv and median_earnings.json** - The datasets used.
- **us.json** - Dataset needed to draw the map.

[8] 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.

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
            |----   d3-queue.v3.min.js
            |----   topojson.v1.min.js
        |--- Q1/
            |----   ...
        |--- Q2/
            |----   ...
        |--- Q3/
            |----   ...
        |--- Q4/
            |----   ...
        |--- Q5/
            |----   ...
        |--- Q6/
            |----   ...
        |--- Q7/
            |----   ...
```