EECS E6893 Big Data Analytics - Fall 2025

Homework Assignment 3: Data Visualization

Due Friday, October 31st, 2025, by 7:00pm

Assignment 3 guidelines:

Datasets:

Part I: “seattle-weather.csv”

Link: https://github.com/vega/vega/blob/main/docs/data/seattle-weather.csv

Part II: “auto-mpg.csv”

Link: https://github.com/plotly/datasets/blob/master/auto-mpg.csv

Part III: Les Misérables dataset (“miserables.json”) on CourseWorks

Submit a **single** pdf with code screenshots and output screenshots. Properly label all the plots and give titles to all the plots (even if not mentioned explicitly). The plots must be clear and readable.

Also submit **all .html files** for the code written as a single zip folder with name as UNI_Name_HW3.zip. Your submission should be **one PDF file** with screenshots of results and code AND **one zip file** containing all .html files.

Part I: D3 Basis (20 marks)

Problem 1: Static graphs (10 marks)

Create webpages with HTML, CSS and plot various plots with d3.js. Use the **Seattle weather** dataset.

1.1.1 Plot basic histogram for the wind variable using D3, bins = 10. Label the X and Y axis properly.

1.1.2 Change the number of bins to 20.

1.1.3 Plot a pie chart using D3 to show the distribution of the weather variable, properly label the sectors with percentage and annotate each sector with corresponding weather variable.

1.1.4 Plot a line graph using D3 to show the trend of the precipitation variable with respect to the date variable. Label the X and Y axis properly.

1.1.5 Add a paragraph to the webpage using the **<p>** tag to describe the main observations from the above plots (e.g., distribution, count, important understanding).

1.1.6 Use the internal style sheet to change the style (font size, font color, indentation) of the paragraph of the web page you created in 1.1.5.

Homework Submissions: Screenshot of your code and the output result.

Problem 2: Dynamic graphs (10 marks)

Now that we have built some basic static graphs using D3.js. We will be moving on to build some dynamic graphs (interactive graphs). Use the **Seattle weather** dataset.

1.2.1 Use the same histogram from 1.1.1 and add an interactive component such that we can **toggle through the bins size**. (Should be able to select any bin size from range 5-25 with increments of 5)

1.2.2 Add a “Variable” dropdown with options precipitation, temp_max, temp_min, and wind. When the selection changes, re-draw the histogram for the chosen variable (using the same dataset and current bin size). Update the axes labels and title to match the selected variable.

Homework Submissions: Screenshot of your code and the output result.

Part II: Auto MPG (30 marks)

In this part you are going to create a basic webpage using HTML, CSS and Javascript and deploy the web page on Apache web server. You will be working on data preprocessing, data visualization and deploying the website on Apache web server. Use the **Auto mpg** dataset.

Make sure that all preprocessing (counting, grouping, etc.) must be done client-side using JavaScript arrays. Do not use Python or other external tools.

Problem 1: Preprocessing (10 marks)

Use the **Auto mpg** dataset. You need to preprocess data to get required columns for data visualization.

2.1.1 Load the .csv file to a Javascript array. Display the top 5 rows of the Javascript array in the webpage as a table in HTML.

2.1.2 Create a new array to get the count of the number of cars of each model year and display the thus obtained Javascript array as a table in HTML.

2.1.3 Create a new array to get the total number of cylinders for each model year and display the thus obtained Javascript array as a table in HTML.

2.1.4 Create a new array to get the count of the number of cars of each acceleration and display the thus obtained Javascript array as a table in HTML.

Homework Submissions: Screenshot of your code and the output result.

Problem 2: Data visualization (10 marks)

You will visualize the data in **a single HTML webpage** using various plots in D3.

2.2.1 Pie chart (uses array from 2.1.2 — counts by model year)
Show the HTML table from 2.1.2, and draw a D3 pie chart where each slice is a model year and the slice size is its car count. Label each slice with model year + percentage and add a title.

2.2.2 Line chart (uses array from 2.1.3 — total cylinders per model year)
Show the table from 2.1.3, and draw a D3 line chart with x = model year (sorted ascending) and y = total number of cylinders for that year. Label axes and add a title..

2.2.3 Histogram (uses array from 2.1.4 — count by acceleration)
Show the table from 2.1.4, and draw a D3 histogram of acceleration with bins = 10. x = acceleration, y = count. Label axes and add a title.

Homework Submissions: Screenshot of your code and the output result.

Problem 3: Deploy webpage on Apache web server (10 marks)

Now that your webpage is ready with the visualized plots. We will deploy the webpage on the Apache web server. Follow the tutorial for installation and deployment steps.

2.3.1 Install Apache HTTP server

2.3.2 Create a Virtual Host

2.3.3 Host your webpage (webpage from Part II problem 2) on the virtual host. (The webpage must be accessible locally via **http://localhost** after hosting. Include a screenshot showing your virtual-host directory and rendered webpage.)

Homework Submissions: Screenshot of Apache installation, virtual host creation and hosting webpage on virual host. And also screenshots for the webpage output.

Part III: Graph Analysis (25 marks)

In this part you are going to create a webpage using HTML, CSS and Javascript and deploy the web page on Apache web server. You will be working on Graph Analysis. Use the Le Miserables dataset.

The Les Misérables dataset represents character co-occurrences from Victor Hugo's novel. Each node corresponds to a character, and edges represent how often two characters appear together in the same scenes.

3.1.1 Plot a Network graph that shows character co-occurrence in Les Misérables. Make sure that you add labels to each node. (each node should show the name of the character)

3.1.2 It can be difficult to observe micro and macro features simultaneously with complex graphs. Use fisheye distortion for the above network graph. (For fisheye, you can use any D3-compatible fisheye implementation that works with your D3 version, or implement a custom fisheye effect by hard-coding).

3.1.3 Add a <p> tag to your HTML file and infer what you see from the graph. For example, you can mention insight about node connectivity or central characters based on the network structure.

3.1.4 Apache HTTP server: Host your webpage on the virtual host.

Homework Submissions: Screenshot of your code and the output result. (Also include a screenshot showing your virtual-host directory and rendered webpage.)

Part IV: Finance data visualization (25 marks)

*This part uses **polygon API** (from previous assignment) to fetch data and display data dynamically on a webpage.*

Problem 1: Finance data visualization from Polygon API (10 marks)

Create a webpage using HTML, CSS, and JavaScript to display **real-time** stock price updates for any three stocks of your choice (e.g., AAPL, GOOGL, MSFT).

4.1.1 Fetch the latest (current) stock price and the previous close using the Polygon API.

4.1.2 Compute and display the difference between the current and previous values.

4.1.3 Use color coding to indicate increase (green) or decrease (red).

4.1.4 Set the page to refresh or fetch new data periodically (for example, every 30 seconds) without reloading. Each update should append a new line of results to the page (do not replace existing values).

If your Polygon plan blocks same-day endpoints, you may use the **latest available** price returned by your plan together with the previous close; display the timestamp you received and still append rows periodically. (Do not hard-code values.)

Homework Submissions: Screenshot of your code and the webpage showing at least three update cycles and color-coded output.

Problem 2: Network Graph for Companies (15 marks)

Build a correlation network for six companies: IBM, Apple, Google, Intel, Microsoft, and Nvidia.

4.2.1 Fetch the closing prices of each company for the last trading day of 2022, 2023, and 2024.

4.2.2 Compute the **Pearson correlation coefficient** between IBM's closing price sequence and each other company.

4.2.3 Plot a network graph in which IBM is at the center and each company is connected to IBM:

Edge distance $\propto (1 - |\text{correlation}|)$
Edge label = correlation value (r)
Node color/size optional for better readability.

4.2.4 Run a loop to show graphs for the three years sequentially on one page (no reload).

To avoid rate limits, you may first fetch the required data from the Polygon API, save it locally (CSV/JSON), and then load the saved file in your webpage to render the graphs. If your Polygon plan cannot return those exact dates, you may instead use five dates separated by one quarter (e.g., 2022 Q4 end → 2024 Q3 end). You must list the five dates used in your PDF, and then compute the network using those values (same correlation and edge labeling rules).

Homework Submissions: Screenshot of your code and the graphs/plots, and a brief paragraph interpreting the correlation patterns.