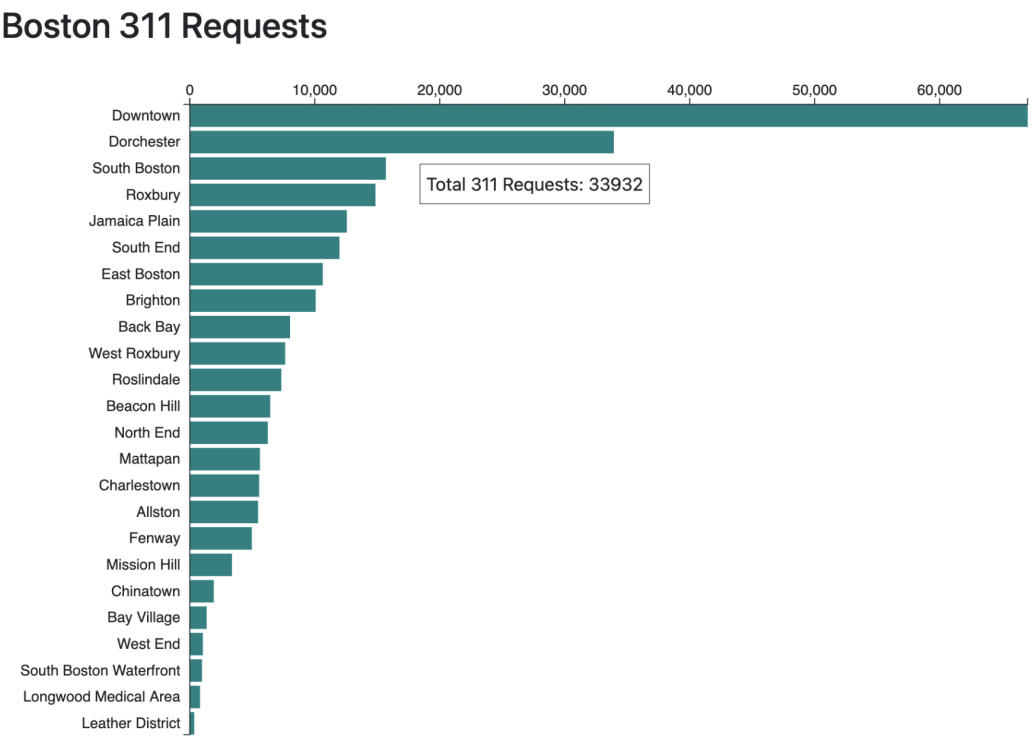
PSET 8: D3 Visualization with External Data

In the last few classes, we've learned more about D3 and how to load external data, create interactive tooltips, and implement scales and axes. In this problem set, we will apply what we've learned to creating a visualization from scratch that includes:

- **External CSV data:** Setting up your own web file structure and loading in external CSV data.
- **Dynamic Axes Scales:** Implementing dynamic scales in D3 to help style chart elements and improve legibility of your data visualization
- **Tooltips:** a tooltip on hover will allow the user to interact with chart elements to view details on specific neighborhoods

Task #8: Visualize Boston 311 Data

This visualization will illustrate Boston 311 data as a horizontal bar chart that displays the total counts of 311 requests per neighborhood. We have provided the boston_311.csv file and will include some requirements and guidelines for how to set up your HTML, CSS, and JS, but we are challenging you to set up your website entirely on your own this time! Your web page end result should look something like this:



Task 8.1: Set up your HTML File

Set up your HTML file and be sure to include the following:

- Include a `<title>` tag with the title “PSET8”
- Include a `<link>` tag to reference the Bootstrap CSS library
- Include a `<script>` tag to reference the D3 library
- Your preference to include embedded CSS and JS within your HTML document or link to an external stylesheet and external javascript.
- Include an element with `class="container"` for a Bootstrap container element on your page
- Include at least one div with `class="row"` for a Bootstrap row element on your page
- Include the headline text “Boston 311 Requests” in a `<h1>` or `<h2>` headline tag
- Create a `<div id="bar-chart">` for where your D3 visualization will go
- Create a `<div id="tooltip" class="hidden">` for where your tooltip will go

Task 8.2: Set up your styles with CSS

Set up your styles with CSS. Be sure to include the following:

- Give the `#tooltip` element the following styles:
  `position: absolute;` /* allows the tooltip to be positioned within the svg */
  `padding: 5px;`
  `background: white;`
  `border: solid gray 1px;`
  `z-index: 1;` /* allows your tooltip to be visible on top of the svg */
- Style an element with both the `id="tooltip"` and `class="hidden"` a style:
  - Use `#tooltip.hidden` (with no spaces between) to select an element with both the the `id="tooltip"` and `class="hidden"`
  - Assign this element the style `display: none;`
- Include styles for the axes ticks (details below in Task #8.3.2 & Task #8.3.3)

Task 8.3: Set up your chart with javascript and D3

Task #8.3.1: Set up your chart with javascript and D3. Define initial variables, set up the svg, load external data, and plot your chart with the data. Be sure to include the following:

- Define and use variables for the `width`, `height`, and `margin` to define the dimensions of your svg. To help with the dimensions, we have used the following values for the answer key plotted above (feel free to use these values as a starting point or customize these values as you see fit):
  - width = 800,
  - height = 600,
  - margin left = 300, margin top/margin right/margin bottom = 30
- Make sure your svg is defined with margins on all sides (you can customize the values to your preference). You can use either the `.attr("viewBox")` or

`.attr("translate", "transform(x, y)")` methods to help position your chart and add margins around the edges
- Use `d3.csv().then(function(data) {})` to load and parse the boston_311.csv file as external data.
- Use the D3 scales and axes methods to plot your bar chart rectangles and axes.
- **Bonus Point for sorting your data in javascript from highest number of 311 requests to lowest number of 311 requests (as shown in the example above)

Task #8.3.2: Set up the x scale and axis. Here are some tips for getting the x values, scaling data along the x-axis:

- First create an array of only the `xValues` for the `total_count` column of data, we can use javascript to *map* values from an array of objects to a simple array using this syntax:
  ```
  const newArray = dataArray.map(d => d.columnName);
  ```
  So to get the x values from our data, try this:
  ```
  const xValues = data.map(d => d.total_count);
  ```
  (and also try testing what the array for `xValues` looks like by using `console.log()` to print out the variable)
- Second, define your `xScale` with `d3.scaleLinear()` to scale your numerical data
  - the domain should go from 0 to the maximum value of `xValues`. Use `d3.max()` to dynamically find the maximum value for the range.
  - the range should go from 0 to the width of the svg
- Plot your x-axis with the ticks located at the top of the chart and specify to only have `6` ticks along the x-axis for every 10,000 requests
- Style a few elements of the axis with CSS:
  - style the text size of all the ticks in CSS to have the `font-size` of `14px`

Task #8.3.3: Set up the y scale and axis. Here are some tips for getting the y values, scaling data along the y-axis:

- First create an array of only the `yValues` for the `Name` column of data, we can use javascript to *map* values from an array of objects to a simple array using this syntax:
  ```
  const newArray = dataArray.map(d => d.columnName);
  ```
  So to get the x values from our data, try this:
  ```
  const yValues = data.map(d => d.Name);
  ```
  (and also try testing what the array for `yValues` looks like by using `console.log()` to print out the variable)
- Second, define your `yScale` with `d3.scaleBand()` to scale the categorical names of

each neighborhood
  - the domain should include all of the `yValues`.
  - the range should go from the height of the svg to 0 (remember in T13 Scales and Axes how the height gets inverted for y values similar to T11 Intro to D3 where we needed to plot height - value in order to flip our chart to the correct orientation)
  - specify an inner padding of `0.15` to give a gap between the bars and labels
- Plot your y-axis with the ticks located on the left of the chart
- Style a few elements of the axis with CSS:
  - style the text size of all the ticks in CSS to have the `font-size` of `14px`
  - hide the tick lines along the y-axis with CSS with the property `display: none;`

Task #8.3.4: Create the rectangle elements for the horizontal bar chart. Here are some tips for getting implementing the scales to plot your data dynamically:

- Include the attributes `x`, `y`, `width`, `height`, and `fill` to define your rectangles
- The `y` attribute can be defined dynamically using the `yScale()` function and the data `Name` for each neighborhood
- The `width` attribute can be defined dynamically using the `xScale()` function and the data `total_count` for each value
- The height attribute can be defined dynamically using `yScale.bandwidth()` to set the height for each bar
- The example above uses the `fill` color teal, but feel free to use a different color if you prefer.

Task 8.4: Add a tooltip

Create a responsive tooltip that displays the data for the counts of 311 requests per neighborhood on hover.

- Create a `<div id="tooltip" class="hidden">` in your HTML for where your tooltip will go
- Add `.on("mouseover", function(event, d) {})` to trigger an event when the cursor enters a rectangle.
  - remove the class `hidden` from the `#tooltip` div
  - give the tooltip a position `left` and `top` that responds to the cursor position with `event.pageX` and `event.pageY`
  - update the text inside the tooltip with the `total_count` data
- Add `.on("mouseout", function() {})` to trigger an event when the cursor leaves a rectangle
  - add the class `hidden` to the `#tooltip` div

Submission

Save your files, host your website live (via GitHub Pages) and submit the URL on Canvas.