TNM048—Information Visualization

Interactive Visualization

January 14, 2018

Kahin Akram Hassan
kahin.akram.hassan@liu.se

1 Introduction

In this lab you will implement parts of commonly used Information Visualization techniques: a Scatter Plot, a Parallel Coordinates plot, and a Choropleth Map to visualize Better Life Index data.

In order to pass this assignment, functional code as well as your explanations of how it works are required. The laboratory exercise should be performed individually or in a group of maximum two students. After completing all tasks, present your results to the laboratory assistant.

2 The Setup

The lab will be done using Html5 and the D3 version 4.0 JavaScript library. Download the file lab1.zip from the course homepage. The file contains the different files that you will use and integrate with your own code.

3 The Data

The data set used in this exercise is the Better Life Index with high inequality (Link here)

The data set is multivariate and stored in a comma separated value file (lab1/static/data/data.csv).

Task 1:
Once downloaded, extract the compressed folder "lab1" and familiarize yourself with the file structure and the data.

4 Scatter Plot

A Scatter Plot view represents the relationships between two dimensions of a multivariate data set by plotting dots on a two dimensional space, additional data dimensions can be mapped on the size and/or color of the dots.The Scatter Plot file sp.js can be found in lab1/static/js/ folder. The scatter plot object is instantiated in the file main.js in lab1/static/js/.

Task 2:
Create 4 variables (x,y,country,circle_size) and assign different data attributes to them in order to visualize and define the scale of the scatter plot axes using the operator `domain()`.

Task 3:
Now add the two axes x,y as well as the two axes titles onto the svg in the method. Use `.attr()`, `.call()`, `.style()` `.call()` `.append()` `.text()` to customize it as you wish.

Now you should be able to see axes for the Scatter Plot with the titles.

Task 4:
Now that we have the two axes with the titles, it's time to add the dots. Start by selecting all the dots from svg and assign it to a variable `circles`. Then use `.data(), .attr(), .style()` to make them visible. Don't forget that a circle has cx, cy, and r as attributes.

Task 5:
You should now see the dots on the scatter plot. One last thing is to add a brush to select one or several dots. Create a variable and call i brush then use `.on()` to call `highlightBrushedCircles` function on this brush and then append `g` div to the svg and call the brush variable. Don't forget to add a class attribute `non_brushed` to the circle variable created in task 4

5 Parallel Coordinates

A Parallel Coordinates plot shows all dimensions of the multivariate data set at once by mapping the data values onto several vertical axes each representing one dimension.
Modify the file pc.js in the folder lab1/static/js/ to implement a parallel coordinates plot.

Task 6:
Use `.scaleBand()` to scale a variable x for the x-axis and use the `dimensions, width` variables to set the domain and range. When done remove the comment the last line on `background,foreground`

We will now add the vertical y-axes to the graph.

Task 7:
Add `dimensions` as data to the variable `axes`. Use `.enter() .append()` to create a `g` div and `.attr(dimension)` as class. Finally use `.attr('transform')` to translate the axes.

and add colors to the lines

Task 8:
Use the `cc` array to fill different colors for the line paths. Create a variable `color` and assign it to `.schemeCategory20` by using a ordinal scale. Then loop over the `data` variable and fill the `cc` array with the colors from the `color` variable. Use the array to color the `foreground` by using `.style()`

Task 9:
Use `.each()` to loop through all the axes and add a `.brushY()` by using `.call()`. Then add `.extent(), and start,brush and end` by using `.on()`

If all the previous steps have been completed correctly you should be able to see the lines and the axes of the Parallel Coordinates. And brush each axes.

6 Choropleth Map

In a choropleth map each region or bordered area is coloured according to a corresponding value in the data set. Edit the file map.js in lab1/static/js/ folder.

Task 10:
Create a color scheme by using orinal scale and `.schemeCategory20`

Task 11:
Create a geoMercator projection as a variable, center it to [60,40] and scale it to (120). Then create a path by using `.geoPath()` and use the projection created above. Then add the path to `country` variable.

Task 12:
Loop through the data with `.forEach()` in order to fill the `cc` array with different colors by using the color scheme created in task 10

7 Brushing and Linking

Brushing means selecting a subset of the data items, usually, in order to highlight them. In parallel coordinates this implies selecting a range of values on one of the axes (dimensions). A brushing technique that toggles the foreground lines of the parallel coordinates is already implemented.

Brushing is most interesting in connection with linking. The Linking technique connects the selecting behaviour between the different views of an application so that selections made in one view are reflected in all others. For instance by brushing lines in a parallel coordinates view, the brush effect (highlighting, etc.) should be applied on those points in the other views that represent the same data items.

Task 13:
There are methods implemented in each file called `selectLine()`, `selectDot()` and `selectCountry()`. Implement two or all methods so that when selecting in one graph the other or others get selected. Remember that calling these methods happens on different places in each file.

8 Tool-tip

A tool-tip is a common graphical user interface element used to display information about an item that is hovered with the mouse pointer.

Task 14:
In map.js and pc.js there is code to show this tool-tip, spend some time to understand how this works.