4DV801: Applied Information Visualization Assignment 1 (jb223qe)

Since I have a background in computer science, I chose to work on task 1 in this assignment. To implement this task, I used javascript and d3 to create the visualizations for both dataset A and B. The following sections will describe my decisions and motivations for the implementation of the two visualizations.

Dataset A

This is a small dataset with a very simple hierarchy. In a tree representation of this dataset, all leaves have a depth of 2. This means that our treemap will not have to visualize any complex, nested nodes.

Overview

Since all leaves have the the same depth and the depth is only 2, a good way of visualizing this in a treemap is to only make rectangles for the leaves and to categorize them all depending on what type of disaster it is (oil spill, forest loss, floods, etc.). We can then color code the different categories in order to differentiate them. In this case, I decided to include a legend to identify the categories. To add some more information to the overview of the treemap, I decided to add a label inside each rectangle displaying the name of that node. In order to avoid overlapping, I made the text visible only if its width was smaller than the width of the rectangle.

Detail on demand

As stated previously, this dataset does not have much detailed information. Therefore, the detail on demand part of the visualization will be very basic. When the user hovers the mouse over a node, a text will appear under the treemap showing the name and value of that node.

Dataset B

Compared to dataset A, this dataset is a bit more complex. The leaves do not all have the same depth and the depth is more than 2 for most leaves. The nodes of this dataset also have some more attributes compared to dataset A. I think a treemap will be suitable for this dataset as well. Some other techniques will have to be introduced in order to convey all information in an appropriate way.

Overview

Since this is more of a nested hierarchy, we will be displaying all nodes of the hierarchy in the treemap. We will also add some outer padding to all rectangles to emphasize that some nodes exist inside other nodes. In this dataset, we will color code the different types of spending. We will also make the color slightly transparent. By doing this, nodes with more depth will appear to have a darker color, emphasizing the hierarchy more. Lastly, labels will be added to the rectangles in the same way as in dataset A.

Detail on demand

In the same way as in dataset A, information about the nodes will be displayed under the treemap when a node is hovered over. This information will include the name, description, source name and source link. The user will not be able to click on the source link but by pressing the enter key when hovering over a node, the user will be redirected to the source page.

There is also some interaction implemented in order to explore the hierarchy closely. By left clicking a node in the treemap, the root data of the treemap will be changed to that node. This gives a “logical zooming” effect. If a node includes many different nodes that are too small to get any detailed information from, a user can click the node to get a bigger view. When the user right clicks, the root data of the treemap will change back to the original dataset.

Interaction discussion

The attributes of this dataset are name, size, description, source name and source link. These are mainly non-quantitative values and can therefore only be visually represented as categories. There does not seem to be any categories among these attributes however. In other words, it would make the most sense to display the information in plain text. The main interaction we can use to display plain text information is that the user selects what node they want to view, and the information gets displayed. This is implemented in my visualization by mouse hovering.

Other than attribute related, a treemap with complex nesting can be difficult to visualize completely in overview due to the potential difference in size between the different nodes. Therefore, logical zooming works very well in this case. An interaction like this is also

implemented in my visualization but it could be made more advanced with smooth transitions and the ability to go back and forth through different depths.