IAT 355
Kickstarter
Brief Report

Assignment 4

Jarielle Lim
Joshua Sun

The audience for the visualization (who and why?)

For our data visualization, the targeted audience for it is towards entrepreneurs and creatives that are interested in starting up a business but do not have the funds to do so and have decided to look to Kickstarter to help back their ideas. Our audience level of data and vis literacy/experience is not high and may only have prior experience viewing business graphs such as graphs to see revenue generated etc. The reason we targeted these types of people is in hopes to discover the most efficient way to start off their Kickstarter campaign with a higher chance of success rate.

The answers or inquiries the visualization is meant to support

The answer that our data visualization is meant to support to our targeted audience is which type of projects is the most popular amongst canadians on Kickstarter and what is the likeliness of it succeeding if one were to start a campaign to raise funds which is represented through number of projects that fail or succeed within different categories.

How the data are mapped to appropriate visual features

For our y-axis, we assigned the main categories in order to indicate the different project types that can be created on Kickstarter which we retrieved from the main_category column from the .csv. For our x-axis, we assigned the number of projects, but the maximum range is only based upon the largest number for the amount of projects in a category from our dataset in order to minimize the amount of white space since the number of projects within the dataset is around 8000 and the category with the most amount of projects is only 1107. Then for the bars, they represent the number of projects in each category and are assigned to a success state which appear as green and failure state which appear as magenta. When you hover over the bars, it will indicate the amount of projects that appear within each state.

Visualization idiom chosen and why

The visualisation idiom we chose to use is the stacked bar graph (comparison and composition) because we were using multidimensional tables with 2 different keys (success/failure and categories) and wanted to compare the part-whole relationship of the success/failures in each category as well as see which category is the most popular for kickstarter projects.

Code Design Brief

We have several variables that utilize d3.nest() in order to help determine values for specific key types since the data we were parsing was being accessed only through one column (main_category).The variable **projectCount** helps gather the number of value to find out what each categories have a specific state and how many times they appear. **mainCatCount** helps find out the amount of projects that appear within each category. Then we did a push array in order to store the values or key within a separate array variable such as **catValueArray** and **catArray**.

To set the x-axis, we took the value from a variable called **maxNumberProject** which held a d3.max function that found the max value of catValueArray which gave us the maximum number of projects in Kickstarter. The y-axis was created by using **catArray** to list the 15 categories available in the main_category column.

To determine the width of the bar for our graph, we made the width attribute of the bar to use the nested data from **projectCount** by calling it using d.values[1].value which essentially gets the key (technology for example) and the nested values under it ("1" is successful and "0" is failure) to determine the length of the 2 bars. In order to make our bar graphs stack, we first added the first bar, then for the second bar, we calculated the starting point by changing the x attribute to where the first bar ends by calculating the successes (also a function calculating d.values[1].value).

colorScale helps determine the range of colours we use in our data visualization, which in our we go from green to magenta (utilising an outside library to create). To make the tooltip and the legend, we used an outside reference to get it to work in our code and implemented an outside library for the legend to make it easier for us to create.

SOURCES

Tool-tips

https://bl.ocks.org/ayala-usma/d2f3b89c84e4ed66e22d02affcdcab73

Nesting with d3.nest

https://bl.ocks.org/ProQuestionAsker/60e7a6e3117f9f433ef9c998f6c776b6

https://stackoverflow.com/questions/27347617/how-to-use-nest-and-rollup-functions-in-d3-to-create-a-bar-chart

Accessing nested data

https://stackoverflow.com/questions/11922383/access-process-nested-objects-arrays-or-json

Legend

http://d3-legend.susielu.com

Scalable

https://stackoverflow.com/questions/16265123/resize-svg-when-window-is-resized-in-d3-js

Reference Libraries

D3: https://cdnjs.cloudflare.com/ajax/libs/d3/4.2.3/d3.min.js

Scale color: https://d3js.org/d3-scale-chromatic.v1.min.js

Legend: https://cdnjs.cloudflare.com/ajax/libs/d3-legend/2.25.5/d3-legend.js