COMP 4304 / 6934 – Assignment 5 (11%)

Due: 11:59pm, Mar. 17, 2023

Learning Objectives

You will gain experience with Matplotlib, Seaborn and interactive ipywidgets in this assignment.

Instructions

Download the following data sets from Brightspace for this assignment:

- `diamonds.csv`
- `names.csv`

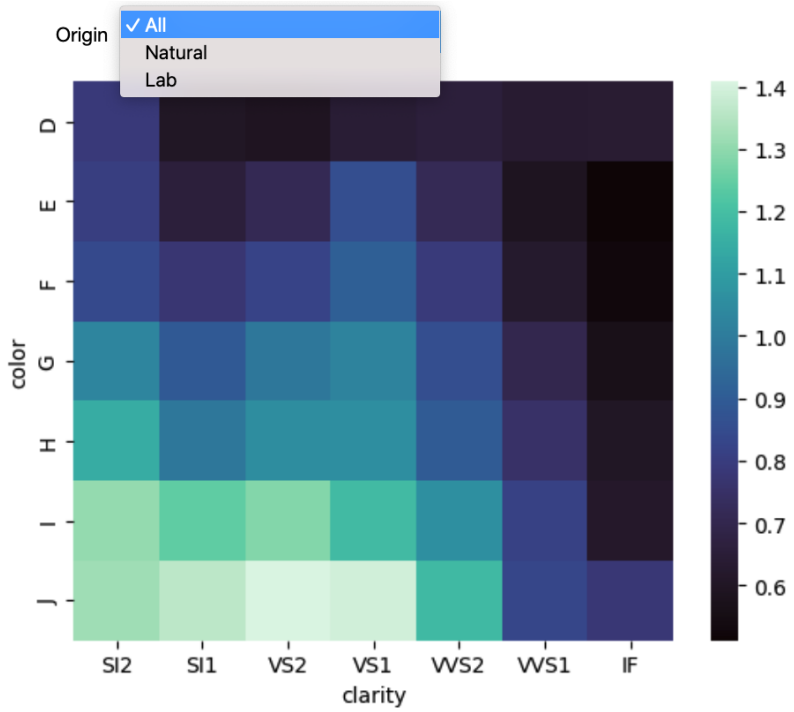
The first data set contains information about diamond sales. Diamonds have a colour, ranging from D (colourless) to J (yellow-ish). Clarity is a measure of any imperfections in the diamond, which go in order from IF (internally flawless), VVS1/2 (very very slightly included), VS1/2 (very slightly included) to S1/2 (slightly included). Carat is a measure of the weight of a diamond. Diamonds are cut to a particular shape and the quality of the cut is measured. Some diamonds are naturally occurring, but some are manufactured.

The second data set contains the rank (popularity) and count of baby names in England and Wales from the period 1996 to 2020.

Using the provided data sets, create a Jupyter notebook that answers the following questions. You may only import the pandas, Matplotlib, Seaborn and ipywidgets libraries.

Question 1: (25 pts)

The following heatmap shows the average carat weight of diamonds relative to their clarity and colour. An interactive dropdown menu allows for the data to be filtered around the origin of the diamonds. The "Natural" option filters to only naturally occurring diamonds, "Lab" filters to manufactured diamonds, and "All' applies no filtering. In the example below, we can see that the heaviest diamonds are those with the poorest quality.

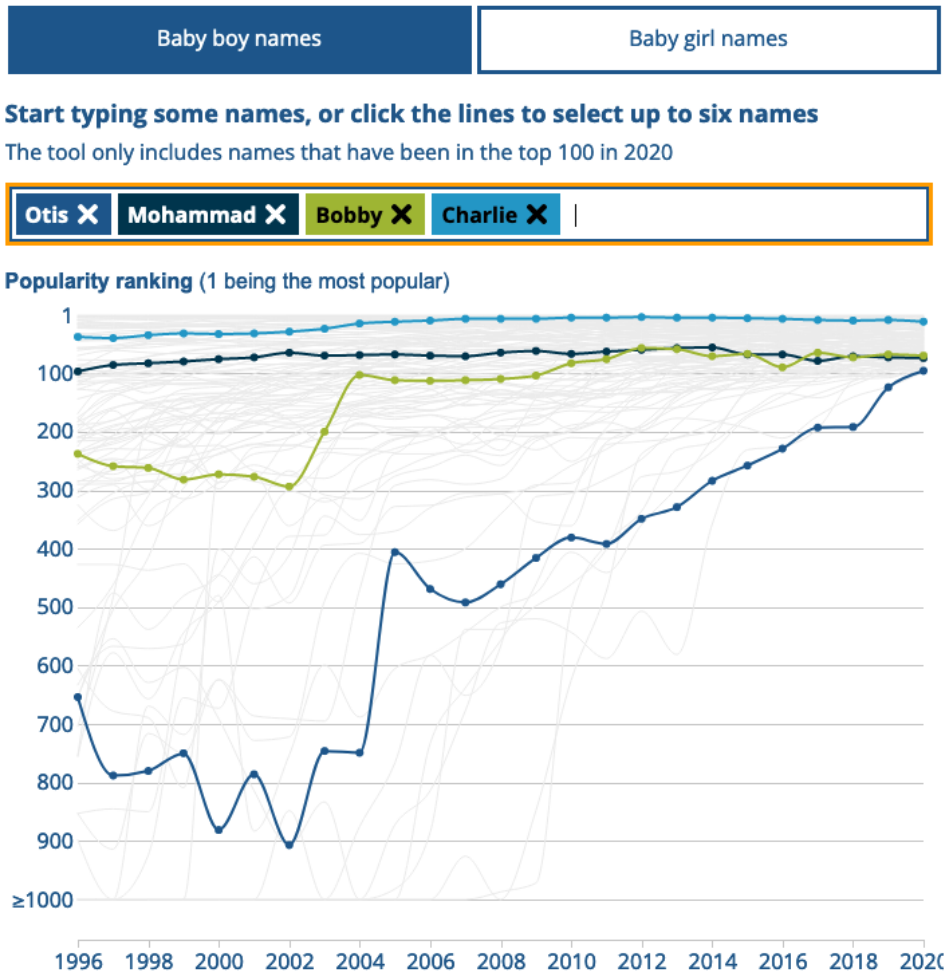


Re-create the above heatmap and interactive feature. The heatmap should update based upon the item selected in the dropdown menu.

The colour map is "mako". The ordering of the colour and clarity should be in ascending order, such that colourless, internally flawless diamonds are in the top right.

Question 2: (75 pts)

Re-create the interactive experience in the "Top baby names" visualization published by the UK Office for National Statistics, which can be found here: https://www.ons.gov.uk/peoplepopulationandcommunity/birthsdeathsandmarriages/livebirths/bulletins/babynamesenglandandwales/2020



This visualization shows the popularity rank of baby boy and girl names over the period from 1996 to 2020. Only the top 100 baby names in 2020 are included. The rank of each of the 100 names is represented with a faded grey line. There is a toggle selection to switch between baby boy names and baby girl names. Names that are specified in the text box will be highlighted.

The interactive features of your visualization should function in the same way as the original visualization. Specifically, they key features that must be included are:

- Toggle buttons that change the represented data between baby boy names and baby girl names.
- A text box where names can be entered. If the name is present, then it will colour that line and add points along the line at each year.
- Multiple names can be entered, separated by a space. Each name should be highlighted.

- Names should be case insensitive.
- A maximum of 6 names can be highlighted.

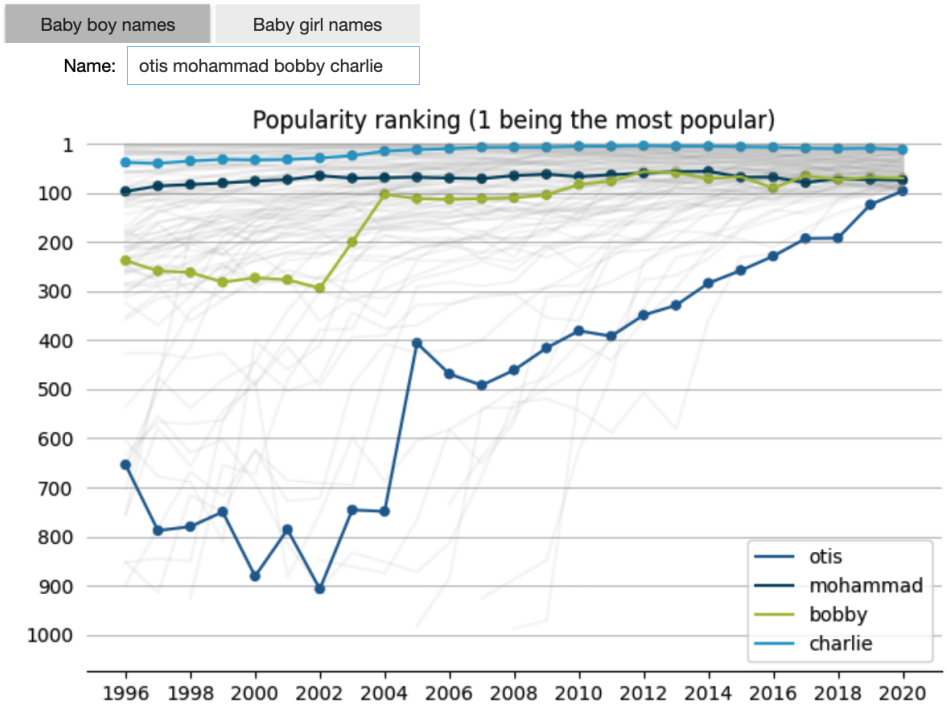
You do not need to match the exact visual look of the interactive elements in the original visualization (which would be hard with ipywidgets). They just need to be present and to function correctly.

Your visualization should generate no warnings or errors.

All 100 top baby names for 2020 should be drawn with a faded grey line. The 6 colours used for highlighting specific names are #206095, #004662, #A8BD3A, #27A0CC, #118C7B, and #F66068.

Other stylistic components should match the original visualization. Each data point should be drawn along highlighted lines. The y-axis should be inverted (hint: ax.invert_yaxis()). The plot title should be present. There should be no labels on the x or y axes. The tick marks on the x-axis should be every 2 years. The tick marks on the y-axis should range from 1 to 1000, spaced on multiples of 100. There should be horizontal grid lines aligned with the y-axis tick marks (hint: ax.grid()). The left, right and top borders should be removed (hit: sns.despine()).

A representative implementation with Matplotlib, Seaborn and ipywidgets is below:



Submission

Submit your Jupyter notebook (`.ipynb`) through Brightspace. Late submissions will be subject to a 10% penalty for each hour past the deadline.

Attribution

Submissions should include an attribution section indicating any sources of material, ideas or contribution of others to the submission.

Submissions must represent your independent work.

You are encouraged to use any resources to help with your solution, but your solution must represent independent work. If your submitted work includes unacknowledged collaboration, code materials, ideas or other elements that are not your original work, it may be considered plagiarism or some other form of cheating under MUN general regulations 6.12.4.2 (4.12.4.2 for graduate students) and academic penalties will be applied accordingly.

Avoid academic penalties by properly attributing any contribution to your submission by others, including internet sources and classmates. This will also help distinguish what elements of the submission are original. You may not receive full credit if your original elements are insufficient, but you can avoid penalties for plagiarism or copying if you acknowledge your sources.

Github

I encourage you to store and version your work on GitHub. It is good practice to do so as everyone uses git in the real world.

However, **it is a requirement that git repositories containing assignment material be private.** University regulations (undergraduate 6.12.4.2 and graduate 4.12.4.2) consider it cheating if you allow your work to be copied. There will be zero tolerance for this.