Data Visualization Concepts

BINF4234

Prof. Dr. Renato Pajarola

Exercise and Homework Completion Requirements

1. Exercises and reading assignments are **mandatory** and they must be completed successfully to finish the class and get a sufficient passing final grade.
2. Exercises are graded coarsely into categories **pass** or **fail**.
   - *A **fail** is given to failed submissions and incomplete solutions, and no points are awarded.*
   - *A **pass** indicates that the exercise is sufficiently good to receive the corresponding points.*
   - *Late submissions (up to one day) will result in "-1" point.*
3. The five exercises give rise to the following point distribution: 2 – 3 – 5 – 5.
   - *A **minimum of 7 points** from all four exercises must be achieved to pass the module. Failure to achieve this minimum will result in a failing grade for the entire module.*
   - *Thus at least two exercises have to be correctly solved, and one has to be from the more advanced ones.*
4. We give **bonus points** for students who have completed more than 8 points from all the exercises.
   - *Thus **7 points** from the exercises is required, **8 points** is still normal passing, and **9 and above** would give 1 or more extra bonus points.*
   - *Only the bonus points can and will be added directly to the final grade.*
5. Do not copy assignments, tools to detect copying and plagiarism will be used.
   - *The exercise results are an integral part of the final course grade and therefore the handed in attempts and solutions to the exercises **must be your personal work**.*

Submission Rules

- Submitted code must compile and run without errors using the indicated Python environment, using the included libraries, packages and frameworks. If additional libraries/packages are needed, please specify in a 'readme.txt' file together with your submission.
- The whole project source code must be zipped and submitted before the given deadline, including the output results (saved in .html file or as a screenshot picture).
- Submit your .zip archive named *dvc_ex1_MATRIKELNUMBER.zip* (e.g. dvc_ex1_01234567.zip) through the OLAT course page.

- **Deadline is Thursday, 15 October 2020 at 23:59h**

Exercise 1

The aim of this exercise is to get familiarized with the basic data manipulation techniques using Python, including data cleaning and basic data visualization. In this exercise, you are expected to visualize the population size distribution across all age groups in Switzerland, grouped by different cantons. You will get access to the online dataset from this link, then complete the tasks as shown below:

Task1: Data Preprocessing.

T1.1: Read the online .csv file into a DataFrame using pandas.

(reference on pandas: https://pandas.pydata.org/pandas-docs/stable/reference/index.html)

T1.2: Prepare data source for plotting: follow the task descriptions in the code skeleton, and generate a ColumnDataSource which contains all the information you need for plotting.

(Make sure that you have an idea of how the data source should be before plotting. You may want to read the reference link in the skeleton first before starting.)

Task2: Data Visualization.

T2.1: Implement the plotting function using the ColumnDataSource as the input, and draw a stacked bar plot grouped by canton and stacked by gender, and try different styling attributes.

T2.2: Add hovering tooltips to the plot in order to provide more detailed information. To be specific, the hover tooltips should display “gender”, “canton, age group”, and “population” when hovering.

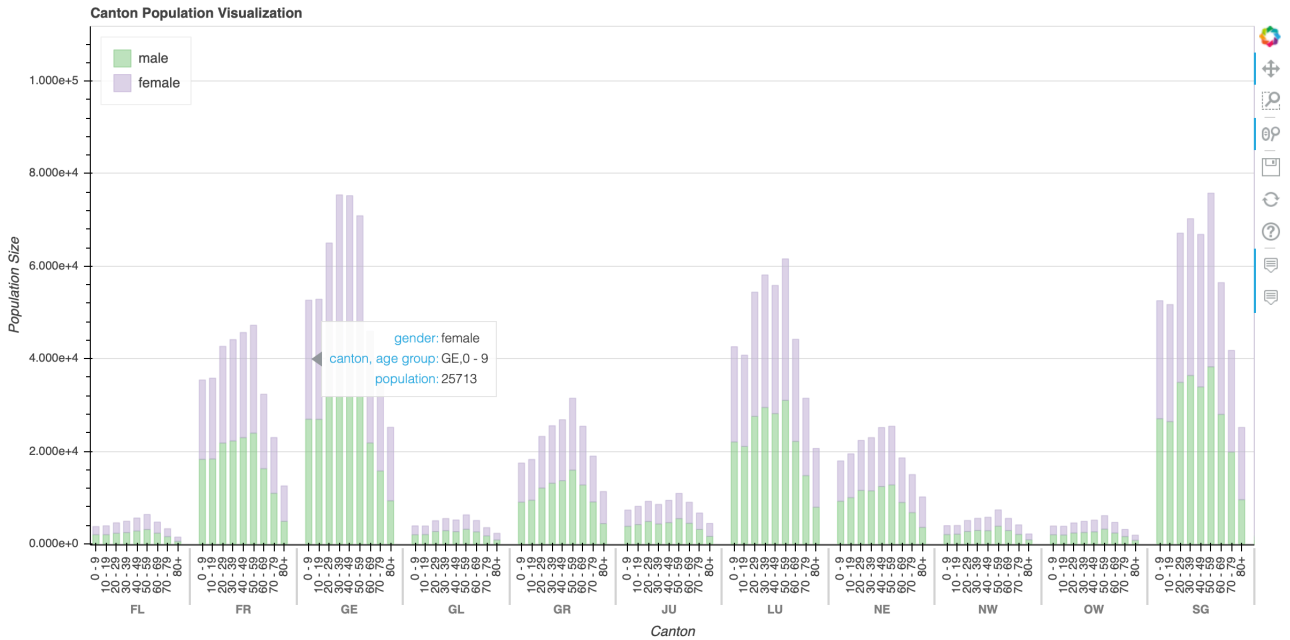
T2.3: Save the plot as a .html file.

Voluntary Task3 (optional): After finishing the first two tasks, you will notice that the subgroup labels are squeezing together which is not aesthetic. To solve this, you can choose one of the solutions below and implement it in your final visualization.

Option1: Choose a subset of cantons from the original dataset and plot the same stacked bar chart as described in task1 and task2.

Option2: Change the subgroup label size. (easier)

The following picture is an example for the desired (but not necessarily the same) visualization result:



Remarks:

- In general, the code skeleton is well structured and divided into groups based on the tasks. However, you may want to change the structure of the skeleton for readability reasons of your own code.
- We recommend to use Jupyter Notebook for your implementation as it can visualize the intermediate output which helps for debugging. However, the final delivery of your code should be .py file rather than .ipynb.
- Try to make good use of the hints and references provided in the skeleton code. (**very important**)
- Try to google first for any Python related issues/bugs.
- Due to the special situation, we don't arrange in person meeting in this semester. Please contact the TA **Fan Feng (fan.feng@uzh.ch)** for technical questions regarding the exercise only if needed.
- More than one day late submission will not be accepted and graded.
- The deliverables of this exercise will be a clean version of your code with proper comments, any additional files necessary for executing it (for example, the data file), a "readme.txt" file for your comments or remarks (if necessary), as well as an export of the final output result in .html or .jpg/.png format. The absence of any required deliverable files will automatically lead to a **FAIL**.