Nadine Merz 16-614-422 | Carolin Carella 16-606-337 | Justyna Urbaniak 16-619-983

Data Analysis with Python: US Baby Names

Concept

This code reads a large dataset and prints the information requested by the user. The user can request answers to 3 main categories about the popularity of baby names in the USA:

1. When their name was the most and least popular

   Input name → Output year
2. The top 10 most common names in a year

   Input year → Output 10 names
3. What name was the most common in a year or in a range

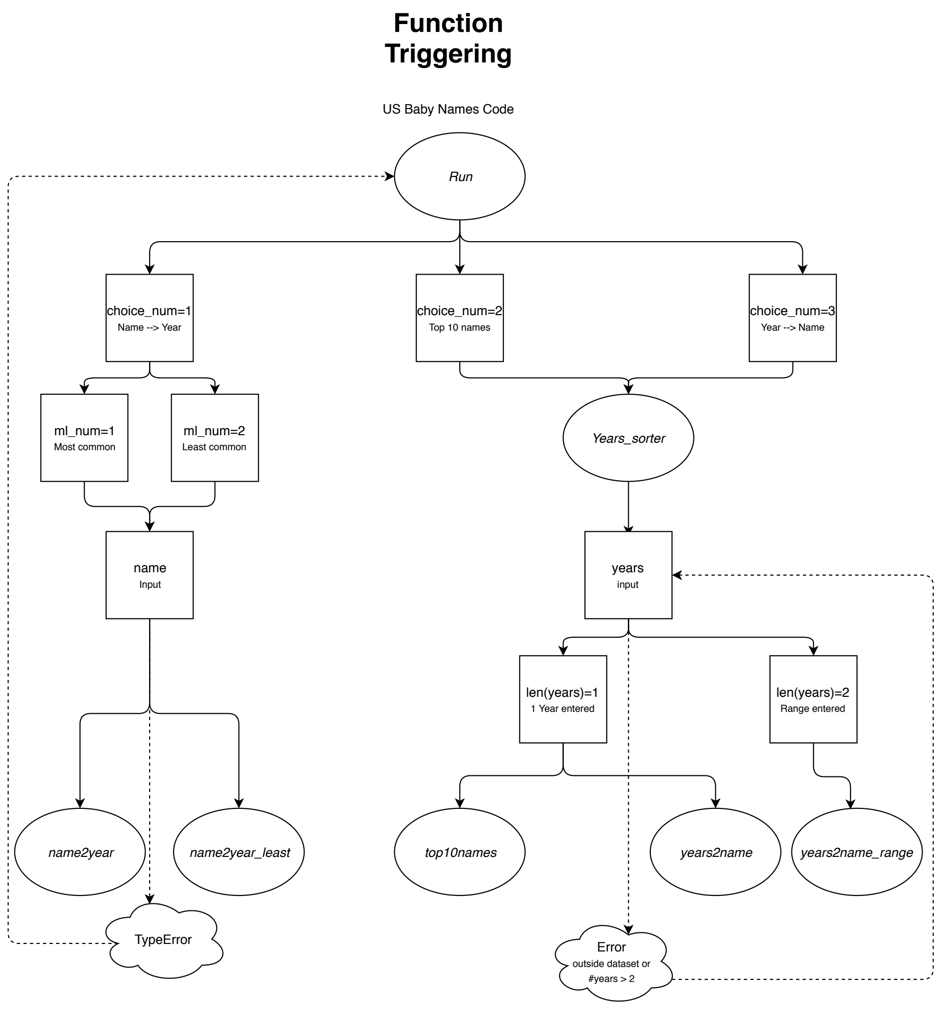
   Input year or range → Output name

Structure and Execution

The code first imports the necessary library Pandas and modules such as time. Then, it opens a csv-file with the columns "Id", "Name", "Year", "Gender" and "Count", which shows baby names in the USA (1880 – 2015) ranked by popularity and divided by gender (download: https://www.kaggle.com/kaggle/us-baby-names). Next, 5 functions are defined, gradually picking out specific lines which fulfil the functional conditions of the input and output variables' relationship until one line is left. The correct column's value is then printed:

1. Functions *name2year* resp. *name2year_least* first create a subset consisting only of lines with a certain name, then scans the subset for the highest resp. lowest value in the "Count" column using max. resp. min. functions and prints the corresponding value of the "Years" column.
2. Function *top10names* creates a subset with lines containing a specific value in the "Years" column and picks out the lines with the 10 largest values in the "Count" column. It then loops through the "Names" column and appends the values into a list which is printed in an enumerated fashion (using the enumerate command).
3. Functions *year2name* resp. *year2name_range* create subsets with lines either with one specific value or a range of values in the column "Year" (using "between"). The subset is scanned for the highest value in the "Count" column same as in 1. With only one line left again, the code prints the value of the "Name" column.

After having defined all necessary functions to print the requested outputs from some input, the code needs to know when to trigger which function. This flowchart summarizes the code structure in this respect:



As category 1 is the only one with a string input, its use is defined in the main function *run* further down. Categories 2 and 3, however, both require integer inputs of a certain range (as dataset only covers 1880 – 2015), so their use is defined in an extra function *years_sorter*. Using a for-loop and several if-functions, it asks the user to input one or two (range) years, and accounts for an input outside the dataset range or a wrong number of values (>2). In both cases, after 2 seconds, the user is brought back to the input option. For an appropriate input, the triggered function depends on the entered number of years (i.e. the length of the input) and the chosen menu option (which is included in the main function *run* further down). For a single entered year (length = 1), choice 2 triggers *top10years* and choice 3 triggers *year2name*. For 2 entered years (length = 2), the user can only want to request function *year2name_range*.

Finally, the main function *run* is defined and subsequently called. The function first prints the menu for the user, letting them input the number corresponding to their desired path. This variable choice_num is defined globally so it could be used in *years_sorter* already. If the user enters 1, now the code defines what to do. Namely, it prints whether the user wants to know in which year a name was most or least common based on their entered value for the variable ml_num. For categories 2 and 3 (i.e. if choice_num is 2 or 3), the code references the *years_sorter* function taking over from here. The code accounts for invalid entries, such as values other than 1 or 2 for ml_num and a TypeError if the checked name in category 1 is not in the database.