Overview

Day	1
Name	Practice #4 Let's explore
Skills	literacy, design, create
Time	1h30 *(finishing in this time line is not required nor graded)*
Submission	No submission

Practice

Preliminary : tools info

Pandas
In computer programming, pandas is a software library written for the Python programming language for data manipulation and analysis. In particular, it offers data structures and operations for manipulating numerical tables and time series.
Pandas is mainly used for data analysis. Pandas allows importing data from various file formats such as comma-separated values, JSON, SQL, Microsoft Excel. Pandas allows various data manipulation operations such as merging, reshaping, selecting, as well as data cleaning, and data wrangling features.
Some library features : DataFrame object for data manipulation with integrated indexing, Reshaping and pivoting of data sets, Data set merging and joining ...

→ Pandas cheatset

Jupyter Notebook
Jupyter Notebook is a web-based interactive computational environment for creating Jupyter notebook documents. The "notebook" term can colloquially make reference to many different entities, mainly the Jupyter web application, Jupyter Python web server, or Jupyter document format depending on context. A Jupyter Notebook document is a JSON document, following a versioned schema, and containing an ordered list of input/output

cells which can contain code, text, mathematics, plots and rich media, usually ending with the ".ipynb" extension.

Seaborn
Seaborn is a Python data visualization library based on matplotlib. It provides a high-level interface for drawing attractive and informative statistical graphics.
Some features : A dataset-oriented API for examining relationships between multiple variables, Specialized support for using categorical variables to show observations or aggregate statistics, High-level abstractions for structuring multi-plot grids that let you easily build complex visualizations ...

→ Seaborn cheatset and tutorials

Preliminary : environment setup

You can either use a local or online environment :

- **Local** : download Anaconda and then follow instructions to install it & other tools (greater details here on notebooks)

  → Tips : on Linux, to set the path, run
  ```
  export PATH=$HOME/anaconda3/bin:$PATH
  jupyter notebook
  ```

- **Online** : https://jupyter.org/try *(think about downloading locally your notebook regularly, otherwise it will not be saved)*

Then, you can upload and open the data and the template notebook available in the practice folder.

Step 1 : Context and data overview

For step 1 to step 3-A included, we will use the 2020 data from the 2020.csv file.

- List files, variables and their type & range (if applicable) in order to have a global picture. You can either open csv in a spreadsheet tool, and/or explore this with pandas.

  → Tips : df.head, df.info, df.describe

- Read a bit about the indicators explanations (page 1)
- Given the variables available, what are the questions that are coming to you ? What

kind of relationship do you anticipate or are curious about ?

Step 2 : Exploring each variable

- Get a better look at each individual variable alone. Check From Data to Viz website to see **what kind of graph you could make** in the decision tree and Seaborn tutorials to see **how to plot** :

  → For one categorical variable *(how many values do they hold each ?)*

  → For one numerical variable *(what do their distribution look like ?)*

  Create charts with seaborn.

  → Tips / sns.distplot & sns.countplot

  What did you learn ?

- For the categorical variable **region**, try to make a vertical bar plot, [Bonus] and a horizontal bar plot

  → Tips / horizontal bar plot : can you find an example that you can reproduce here ?

  What did you observe on the data from the charts ?

- [Bonus] If there are many numerical variables, how could you automate the creation of distribution plots for all of them ?

  → Tips / get all numerical variables :
  df.select_dtypes(exclude=['object']).columns

  → Tips / create subplot with matplotlib :
  fig, ax = plt.subplots(**X**, **Y**, figsize=(**A**, **B**))

  → Tips / iterate over a list of variables, and given subplots :
  for variable, subplot in zip(**VAR**, ax.flatten()):

  → Tips / create a specific plot on a given subplot with seaborn :
  sns.**COMMAND**(df[variable], ax=subplot)

Step 3 : Exploring variables relationships

A) Staying in 2020

- **1 Numerical x 1 Numerical** : what kind of plot can we use to see the relationship between 2 numerical variables ? Make at least one chart.

  [Bonus] Can you find a chart in the seaborn documentation that helps you to see in one chart the relationship between all numerical variables ?

  → Tips / see the Multi-plot grid section of Seaborn tutorials (pairplot)

- **1 Categorical x 1 Numerical** : what kind of plot can we use to see the relationship between one numerical variable and one categorical variable ? Make at least one chart.

- **1 Categorical x 2 Numerical** : what kind of plot can we use to see the relationship between one numerical variable and two categorical variables ?

  For example, how can you visualize the happiness score, the gdp per capita and the region variables ? Make at least one chart.

  If it's not lisible, how could you plot small multiples, e.g. one subplot for each region (the categorical variable) ?

  → Tips / small multiples : FacetGrid
  → Tips / Plot size : col_wrap=**VAR**, height=**VAR**

What did you observe on the data from the charts ?

B) Looking into the past

Here we want to explore further about the evolution of the happiness scores over time. In the notebook template, I already merged data from original files between 2015 and 2020.

- Let's see the evolution of happiness scores for one country over time. What type of charts can show the **evolution** relationship of a variable ?

  Would you consider the 'year' variable as a numerical one ? a categorical one ? how can you show that the score value associated to years are taken at a specific point in time ?
  → Tips / select rows based on values with pandas : df_filtered = df.loc[**condition**]
  → Tips / sns.lineplot sns.pointplot

- Let's plot the evolution for all countries. Is that lisible ?

- Let's try to look at this with 2 different approaches :

1/ Separating countries that have a positive / negative trend. We are defining here a new variable to “add meaning” : trend, which is True if the happiness score in 2020 is higher than the score in 2015. (*This is a limited proxy for assessing a trend, but we’ll use it because it’s simple enough).*

What are you seeing in this chart ? What can you conclude ?

2/ Looking at more specific groups of countries : the most and the least happy. The process on the most happy countries is provided.

Can you do the same with the least happy countries ? Start by choosing a happiness value under which you will select countries.

- Are there other chart types you want to explore ? Other approaches you would have preferred ?

  You can also check : https://www.data-to-viz.com/caveat/spaghetti.html

What did you observe on the data from the charts ?

Step 4 : Finally, to be more critical (15 minutes)

Meet with one person and discuss :

A) Chart uses and findings :
   → Any trouble in this practice ?
   → What did you learn ? What did you find and not find ?
   → Did you do the same charts ?
   → Are there charts you wanted to make and did not have the time ? Which one ? Can you draw them ?
   → Are there questions left where you did not investigate ? Can you imagine/draw chart types & related variables that could be done ?

B) Can we really make firm assumptions from this visual analysis ?
   → https://ourworldindata.org/happiness-and-life-satisfaction
   → https://en.wikipedia.org/wiki/World_Happiness_Report#Criticism