CONSTRUCT MINING PIPELINE WORKBOOK

———

A GENERAL REMARK

If you are stuck with an exercise, you can always:
/ ask me.
/ look up the original repository for inspiration.
/ use the solution cheat sheet.

EXERCISE 1: SENTENCE EMBEDDINGS

1. Import the *SentenceTransformer* class from the *sentence_transformers* package. Follow the link to the sentence transformer documentation, if you need help.
2. Look up the indicated sentence transformer model on its Hugging Face model cart by following the provided link. Import the model.
3. Import the raw strategy sentences as data. Use the *pd.read_csv()* function. The file name is *'data/strategies_raw_en.csv'*.
4. Convert the strategy sentences to a list. You can do so by calling the text column of *data* (*data.text*) and use the *.to_list()* function.
5. Embed the strategy sentences with the sentence transformer model we imported earlier. You can do so by calling the *model.encode()* function. Pass the strategy list to the function. Set the parameter *show_progress_bar* to *True*.
6. Execute the cell with the *np.save* function to save your personal sentence embeddings to a file in the data folder. No code modifications needed.

EXERCISE 2: MEASURING ITEM BIAS

1. Look at the table for the item bias statistic (next page). For which items does a bias exist and in which direction?

Table 1. Item bias statistics before bias reduction.

Item	Bias Statistic	Confidence Interval
1	-0.99	[-1.19; -0.79]
2	-1.79	[-1.91, -1.66]
3	-2.92	[-3.19; -2.67]
4	-1.96	[-2.10; -1.81]
5	-2.11	[-2.26; -1.97]
6	-0.91	[-1.01; -0.80]
7	-0.86	[-1.00; -0.71]
8	-1.23	[-1.37; -1.09]
9	-1.51	[-1.71; -1.29]

EXERCISE 3: REDUCING ITEM BIAS

1. Look at the vignette texts below. For each vignette try to identify words that are related to the situation, but not to potential emotion regulation strategies in response to the vignettes. Circle them. Vignette one is provided to you as an example (words printed in bold).
2. Put all your identified keywords into a dictionary* in the code. The keywords need to be provided as lists of strings per vignette. Separate your entries in the dictionary for each vignette with a comma. The code already provides a minimal example.
3. Execute the whole notebook to create your personal masked embeddings. They are saved as a file in the data folder.

* In this context, a "dictionary" is just the name of the data structure that is used here.

Vignette texts

1) Your workmate fails to deliver an important piece of **information** on time, causing you to fall behind **schedule** also.
2) You are accepted for a highly sought after contract, but have to fly to the location. You have a phobia of flying.
3) You answer the phone and hear that close relatives are in hospital critically ill.
4) You find out that some members of your social sports team have been saying that you are not a very good player.

5) You have just gone back to university after a lapse of several years. You are surrounded by younger students who seem very confident about their ability and you are unsure whether you can compete with them.
6) A demanding client takes up a lot of your time and then asks to speak to your boss about your performance. Although your boss assures you that your performance is fine, you feel upset.
7) Your access to essential resources has been delayed and your work is way behind schedule. Your progress report makes no mention of the lack of resources.
8) You are having a large family gathering to celebrate you moving into your new home. You want the day to go smoothly and are a little nervous about it.
9) You and your colleague usually go to a cafe after the working week and chat about what's going on in the company. After your colleague's job is moved to a different section in the company, he/she stops coming to the cafe. You miss these Friday talks.

EXERCISE 4: REDUCING DIMENSIONALITY AND CLUSTERING

1. Choose any number that you like and set is as *RANDOM_STATE* to initialize our algorithm.
2. Scroll down the notebook, where we call the evaluation_loop function in the second to last cell. Choose 3 different values to vary over UMAP's number of dimensions (also called components) and pass them to the function as list of integers under *Ns* (e.g., [10, 30, 50]). The integers have to be less than 768, which is the size of the original embedding space.
3. Execute the whole notebook.
4. Look at the plot. You will see the number of clusters plotted as a function of different combinations of two UMAP and HDBSCAN parameters: number of dimensions (the three subplots), and k nearest neighbors (on the x axis). Which combination of parameters would you choose to achieve the following goals:
    - Achieving a number of clusters between 30 and 40.
    - Keeping the number of dimensions after projection as high as possible (as indicated by UMAP's n).

EXERCISE 5: CHECKING ROBUSTNESS

Assume we found a set of parameters that satisfies our aforementioned criteria (and some more). Further assume that we ran UMAP and HDBSCAN a thousand times with those parameters, but different random seeds, and saved our evaluation variables to files - among them numbers of resulting clusters. We would now like to find a solution, i.e. random seed, that creates the most prevalent number of clusters.

1. Find out the modus for number of clusters. You can do so by converting the *no_clusters* variable (list) to a *pd.Series()* datatype and apply the *.value_counts()* function. You can verify your solution by looking at the histogram below.
2. Extract a random seed of the thousand tested seeds that produces the modus for number of clusters. You can do so by inserting the desired number of clusters at the indicated spot in the code.

EXERCISE 6: VALIDATING CLUSTERS

1. We are now conducting our own small survey to validate one of the clusters from the final solution. Please access the survey via the QR code or link, and answer the four questions.

www.fbr.io/rnycw

2. Let's now look at the results from the intrusion survey for all clusters. We want to identify the clusters for which less than 75% of intruders (i.e., the majority threshold) were identified. Run the notebook. Complete the plot by adding a horizontal line at 0.75 to indicate the threshold. You can use the *ax.axhline()* function and set its parameter *y* to 0.75. Which clusters fall below the threshold?

EXERCISE 7: INTERPRETING CLUSTERS

1. In the notebook, set the *CLUSTER* variable to one of the following numbers: 4, 5, 9, 17, 20, 23, 28. Run the notebook (it might take a few minutes). You can now inspect the final cluster solution visually. Look at the sentences that are subsumed under the cluster number you chose. Which emotion regulation strategy class do you think the cluster represents? Give it a name.