Lab activity: analysis of music networks.

The first part of the practice (the first 4 sessions) is completely guided and covers the entire data lifecycle through the implementation of a set of functions defined in the exercises and the analysis of the obtained results. On the other hand, the **second part** of the practice is more **open-ended**, and students will be able to apply the knowledge acquired in the first part to propose and solve a data science problem based on the analysis of networks of personal interest.

In this second part of the practice, students will propose a problem or question of their own interest (based on the data provided by the Spotify API) and study or answer it using the graph analysis tools covered in the course. This process will involve the usual phases of data analysis (data acquisition, preprocessing, analysis, and visualization).

Regarding the problem to be proposed, students can choose a specific detail of interest or a broader project (some examples will be described in the following section). In any case, the problem formulation will be evaluated, as well as the techniques used to solve it (in particular, **linking the theoretical concepts covered in the course with their interpretation in the context of the problem at hand**), the adequacy of the analysis to the posed question, and the presentation of the results supported by appropriate visualizations.

Regarding the implementation and data sets, you can use the developed functions and the data obtained in the first part of the practice, as well as create and acquire new ones if you consider it appropriate.

The second part of the practice consists of a single session (session 5).

1 Session 5: Personal Network Analysis Project

Below are some ideas for possible projects that you can develop in this session. You can use these projects (or variations of them), but **we strongly recommend proposing your own projects** that align with your interests.

1.1 Examples

In the first part of the practice, we worked with the graph of related artists provided by Spotify. Some questions or objectives that you can explore about this graph include:

- The objective of this work is to **reverse-engineer Spotify's algorithm for selecting related artists**.

1. Based on multiple instances of the graph of related artists obtained by running the crawler at different time points (e.g., once a day), describe how often the related artists are updated and the volume of change observed. In addition to observing the variation in the number of nodes and edges in the graph, other measures of similarity between the obtained graphs (such as edit distance) can be analyzed to quantify the changes that occur.
2. Using the obtained graph of related artists, attempt to determine if certain artist properties seem to be taken into account when artists are related. For example, evaluate whether the popularity of an artist, the number of followers, the musical genre they are tagged with, or the audio features of their songs are attributes considered when artists are related. Some specific questions you can ask are: Does the genre of two artists need to match for Spotify to relate them? Does the popularity of an artist impact the popularity of the related artists? Is it necessary for two artists to have collaborated on a song for Spotify to relate them?

Beyond related artists and the most popular songs of each artist, the Spotify API contains a lot of other information. Some projects that you can tackle using this information are:

- Spotify's catalog contains tags to define thousands of musical genres. The objective of this work is to **group these musical genres by similarity**. To do this, a genre graph will be created based on co-occurrences in artists: the nodes will represent the genres, and they will be connected by an edge if the same artist is tagged with both genres (the weight of the edge will be the number of times this occurs). Then, a community detection algorithm will be applied to group the genres.
- The objective of this work is to **identify sets of artists that frequently collaborate** in song performances. To do this, a graph of artist collaborations will be created: the nodes will represent artists, and they will be connected by an edge if the two artists have collaborated on a song. The weight of the edge will be the number of songs in which they have collaborated. Then, a community detection algorithm will be applied to identify groups of artists that frequently collaborate.

1.2 Questions to answer in the report

1. Define the objectives of your study. What question or objectives do you aim to address?
2. Describe the process of data acquisition and the obtained data.
   (a) Explain how you obtained the data (e.g., search criteria or crawler scheduling algorithm).
   (b) Which API endpoints did you use to acquire the data?
   (c) Describe the obtained data. For graphs, describe the type of graph used, what the nodes and edges represent, and their attributes. For tabular data, describe the collected entities and their attributes.
   (d) Indicate the volume of collected data. For graphs, provide the size and order. For tabular data, provide the number of records.

(e) Detail the format used to store the data.

3. Describe the data preprocessing tasks.
4. Describe the data analysis tasks.
   (a) What network analysis algorithms did you use?
5. Describe the visualization tasks.
   (a) What tools did you use to generate the visualizations?
   (b) What parameters or configurations did you use?
   (c) What problems did you encounter, and how did you solve them?
6. Answer the initial question using the results of the performed analyses and generated visualizations.

1.3 Evaluation and expected results for this part of the practice

This fifth session of the practice will account for 20% of the total grade for the practice.

Remember that it is important to **include sufficient comments** in the Python code to understand its functionality.

The files to be obtained in this fifth session of the practice for the final submission are the following:

- A file named `Lab_AGX_202223_S5.py` containing all the necessary code to solve the problem posed in this session. This file may include `imports` from other code files developed in the first part of the practice (if you make use of any of the functions).
- A PDF file that provides detailed answers to all the questions raised in this prompt. The answers to the questions should be ordered.
- The obtained data files.