Data to Network: Movies

The objective of this class is to show that meaningful networks can be created from data even where the data does not naturally form links or relations. A network can be used to model correlation/similarity between elements. Of course, when it is possible, it is better to work directly on the original data, e.g., a correlation matrix. But representing it as a network – which means usually using a threashold to keep only the most significant correlation – extend the possibility of analysis. Be careful that some properties might not be significant in such a networ: the degree, for instance, relates to something like being similar to many other elements, rather than being "popular" or "important". But analyzing who is connected to whom, the general organization, Communities, relations between node properties and their connexions, etc. still are relevant to explore. Tools such as link predection or node classification, that we will see later, are also relevant on such a network. Finally, due to the sparse nature of networks, many problems become much more tractable as networks than as full matrices, for large datasets.

1. Getting familiar with the data
   (a) Follow the notebook at https://github.com/Yquetzal/Teaching_notebooks/blob/main/Networks/Movie-similarity.ipynb.
   (b) Create a first custom network by increasing the number of movies (100-200 maybe), and trying different thresholds.
   (c) Visualize/Analyze the results
2. The right correlation
   (a) Following the tutorial, you used Cosine Similarity to compute the similarity between movies. But that is not the only solution. A common variant is to use Pearson Correlation coefficient (https://en.wikipedia.org/wiki/Pearson_correlation_coefficient). Use it to create another network. How different is it ? Which one do you think is the most appropriate?
   (b) Let's think a little about what Cosine Similarity means. Search for the definition of this measure as a sum over the elements of the vectors (https://en.wikipedia.org/wiki/Cosine_similarity). Let's remember something: scores can take values only in a very limited range of values, and most of the time, people use scores as votes: they give mostly good scores, only to movies they like (Implicit rating). Let's assume that you transform the data to binary values: the value is either 0 (if score$\leq$4) or 1 (score>4). Cosine similarity boils down to something very simple in that case, what is it ? Can you think of weaknesses of this score? Think for instance that some movies are very famous, and thus have many votes, while other have little votes, because they have a lower budget, belong to a less popular category, etc.
   (c) Check the definition of the Jaccard Index (Jaccard Coefficient). Do you think it could be more appropriate ? If yes, use it and compare the networks
   (d) People tend to have different notions of what a good score is (Rating Habits). If you used a score threshold at some point, it might be wise to normalize scores first.
3. Designing our own score
   (a) If our problem does not fit typical situations, we might be better off designing our own score. I propose to use the following idea: When (and only when) two movies have been rated by the same person, then, if this person has given a similar score to both, it's a hint that these movies are similar (liked/disliked by the same people). So we could use some *rating difference*, e.g. the average rating difference among all viewers of both movies.
   (b) Think about the potential biases of those score: Implicit rating, Rating Habits, Popular VS unpopular movies... Propose your own score to try to minimize those problems. (No *right* solution !)

4. Sharing
   (a) Share your result and observations with other students.