Exercise set #5 (14 pts)

- The deadline for handing in your solutions is October 23rd 2023 23:59.
- There are multiple ways for submitting your solutions. Check each question for details.
- Return also one `.zip` file containing all your Python code of the round in MyCourses.
- Check also the course practicalities file in MyCourses for more details on submitting your solutions.

1. Degree correlations and assortativity (7 pt)

In this problem, we consider degree correlations and assortativity of two real-world social networks: the Zachary karate club network (`karate_club_edge_list.edg`) [1] and a snowball-sampled subgraph of a Facebook friendship network (`facebook-wosn-links_subgraph.edg`) [2, 3].

Perform the following analyses (you can use the code template provided). Return your solutions by taking a MyCourses quiz.

a) (2 pt) **Create scatter plots** of the degrees of pairs of connected nodes. That is, take each connected pair of nodes (i, j), their degrees k_i and k_j, plot the point (k_i, k_j) on two axes with degrees as their units, and repeat for all pairs of connected nodes. Do this for both the Karate club and Facebook networks. Because the networks are undirected, the plots should be symmetrical, containing points (k_i, k_j) and (k_j, k_i) for all connected pairs (i, j).

b) (1 pt) For the Facebook friendship network, **produce a heat map**[1] of the degrees of connected node pairs. The heat map uses the same information as you used in part a), that is, the degrees of pairs of connected nodes. However, no points are plotted: rather, the two degree axes are *binned* and the number of degree pairs (k_i, k_j) in each bin is computed. Then, the bin is colored according to this number (e.g., red = many connected pairs of nodes with degrees falling in the bin). **What extra information do you gain** by using a heatmap instead of just a scatter plot (if any)?

c) (2 pt) The assortativity coefficient is defined as the Pearson correlation coefficient of the degrees of pairs of connected nodes. **Calculate** and report the assortativity coefficients for the two networks both using `scipy.stats.pearsonr` and the NetworkX function `degree_assortativity_coefficient`. Check that both methods return the same value. As mentioned in the lecture, social networks are typically assortative. **Does this hold** for the Zachary karate club network? **Explain** how the specific structure of the network impacts assortativity in this case.

d) (2 pt) For the Facebook network, **compute** the average nearest neighbour degree k_{nn} of each node and **make a scatter plot** of k_{nn} as a function of k. In the same plot, **draw** also the curve of $\langle k_{nn} \rangle(k)$ as a function of k, i.e. the average of k_{nn} for each k value. In a perfectly assortative network, $\langle k_{nn} \rangle(k)$ increases monotonically as a function of k. **Is this the case here**? Looking at the curve, can you identify different regimes in terms of assortativity? **Describe** them shortly.

[1] `http://en.wikipedia.org/wiki/Heat_map`

2. Community detection (7 pt)

In this exercise, we are going to learn how a simple community detection method works and how different algorithms perform on real-world networks. The network data that we are going to use are:

- Zachary's karate club network (`karate_club_edge_list.edg`): A classic social network of a karate club, originally collected by Zachary in 1970s.
- Dolphin social network (`dolphins_edge_list.edg`): A social network of bottlenose dolphins collected by Lusseau et al. (2003).
- Political blog network (`polblogs_edges.csv`): A network of hyperlinks between weblogs on US politics, recorded in 2005 by Adamic and Glance.

Return your solutions by taking a MyCourses quiz.

a) (3 pt) Your first task is to **implement** the label propagation algorithm. Label propagation is a simple yet efficient algorithm based on the idea that each node is likely to be linked to other nodes belonging to the same community. There are ready-made functions for this in NetworkX, but here we want to implement it ourselves. The algorithm works as follows:

1. Initially, each node is assigned to its own community with a unique label.
2. In each iteration, each node adopts the label that is most frequent among its neighbors. In particular:
   i) All nodes are visited in a random order.
   ii) When a node is visited, it updates its community label to the most frequent label among its neighbors. If there are several such labels, one of them is chosen randomly.
3. If every node is assigned a community label that is most frequent among its neighbors, no more update is possible and the algorithm stops. Otherwise, it repeats step 2.

Let us use the implemented algorithm to **find communities** in the karate club network. Since the label propagation algorithm is non-deterministic, the communities found may vary between runs. One way to deal with this issue is to run the algorithm multiple times and adopt the most frequent community structure as the result. We know that a conflict between the karate club members during the study caused the club to split into two. So, the karate club network has two communities. Therefore, let us **sample 1000 runs** that find two communities (bipartitions) and take the most frequent bipartition as our guess for the community structure of the network. **Visualize** the network with nodes colored according to the communities found.

Hints:

- A random order can be obtained by using `rng.permutation()` or `rng.shuffle()`.
- In case of a tie, `rng.choice()` can be used to choose one at random.
- You can also use `collections.Counter()` to count the frequency of community labels among the neighbors of a node.

b) (2 pt) Next, let us compare the performance of different community detection methods. There are many different algorithms out there, but we will be focusing on the following three:

- Infomap: A random walk based algorithm which tries to find a partition that minimizes the description length of the trajectory of the walker.
- (Semi-synchronous) label propagation: Similar to the one you implemented above, but uses a synchronous updating scheme.
- Louvain algorithm for modularity maximization: A greedy algorithm that tries to find a partition that maximizes the modularity of the network.

Your task is to **apply these three methods** to find the communities in the dolphin network and the political blog network. Here again, Infomap and Louvain are non-deterministic algorithms that may generate different partitions every time they are run. To mitigate the effect of randomness, **run each algorithm** 100 times for the dolphin network and 20 times for the political blog network, and choose the partition with the shortest code length (Infomap) or highest modularity (Louvain) as the result.

Report the number of communities and modularity of the partition found by each algorithm. In addition, **visualize** the partitions for the dolphin network. Then **select all** the correct statements from the following:

A) Infomap and the Louvain algorithm find similar community structures with similar modularity values in the dolphin network, but label propagation finds a different one.

B) All three methods find a similar community structure in the political blog network.

C) Infomap and the Louvain algorithm find different numbers of communities in the political blog network, but the modularity values of these partitions are similar.

D) In these two networks, label propagation does not find a community structure that maximizes modularity.

Hint: This time, you may use ready-made functions in NetworkX. Read the NetworkX documentation for community detection algorithms. We will use the implementation of Infomap in the `infomap` package, as the implementation is not included in NetworkX.

c) (2 pt) Finally, **generate** an Erdős-Rényi network with $N = 500$ nodes and average degree $\langle k \rangle = 10$, and **apply** the three community detection methods to it. Then **select all** the correct statements from the following:

A) The Louvain algorithm outperforms the other two methods in finding a partition that maximizes the modularity.

B) The fact that the Louvain algorithm finds multiple communities in the Erdős-Rényi network suggests that maximizing modularity is a bad idea for finding communities in random networks.

C) Label propagation is not a good method for community detection in this case, as it can only find a partition with a very low modularity value.

D) The fact that Infomap finds the entire network as a single community while maximizing modularity is proof that Infomap is a better community detection method than the other two.

Hint: If the generated network is not connected, regenerate another one until you get a connected network.

Feedback (1 pt)

To earn one bonus point, give feedback on this exercise set and the corresponding lecture latest two days after the exercise round's submission deadline. You can find the feedback form in MyCourses.

References

[1] W. W. Zachary, "An information flow model for conflict and fission in small groups," *Journal of anthropological research*, vol. 33, no. 4, pp. 452–473, 1977.

[2] [Online]. Available: http://konect.cc/networks/facebook-wosn-links/

[3] B. Viswanath, A. Mislove, M. Cha, and K. P. Gummadi, "On the evolution of user interaction in facebook," in *Proceedings of the 2nd ACM workshop on Online social networks*. ACM, 2009, pp. 37–42.