CS 506 Spring 2020 - HW3

Social Networks and Recommendation Systems

Due date: April 27, 2020

1 Background

In this homework, you will try to recommend new collaborations to researchers of the Machine Learning community. Our approach will follow the guidelines of collaborative filtering: “If your past behavior/preferences were similar to some other user’s, your future behavior may be as well”. As an example, imagine you like Rolling Stones, Beatles and Jimmy Hendrix. It turns out that most people that like the aforementioned artists, are also fans of Eric Clapton. Then, it is very likely that if you listen to Eric Clapton’s music, you will like it as well.
In this assignment you will implement a collaborative filtering recommendation system for suggesting new collaborations to Machine Learning researchers.

A network as a graph: A graph or network represents relationships among different entities (users of a social network, researchers, products, etc.). Those entities are represented as nodes and the relationships between them (friends on Facebook, co-authors of a research paper, products purchased together) as edges. When there is an edge between two nodes, x and y, we say that y is a neighbor (or friend) of x (and also - as the graphs we consider are undirected - x is also a neighbor of y).

Representing a graph in Python: A widely used library in Python, for representing graphs is NetworkX. You can read the documentation for more information on how to use this library.

2 Recommend new collaborations - The ML Community case

In order to provide new collaborations and test the efficiency of the methods used, you are given two files (you can find them on piazza):

- "old_edges.txt": In this file, every line contains the names of two researchers that have co-authored a paper in one of the top Machine Learning conferences (NeurIPS, ICLR, ICML) between 2010 and 2016.

- "new_edges.txt": In this file, every line contains the names of two researchers (from those existing in the above file) that formed a new (non-existing before) collaboration, in either 2017 and 2018.

With the first file in hand, you will answer the following question: "For author X, list some non-collaborators in order, starting with the best collaborator recommendation and ending with the worst". A non-friend is a user who is not X and is not a collaborator of X. Depending on the recommendation algorithm you are going to choose, the list may include all non-collaborators or some of them.

Then, using the second file, with actual new collaborations formed in the next 3 years, you will test the efficiency of these algorithms.

3 Tasks

a) [3 pts.] Write a function that reads the file "old_edges.txt" and create a graph using NetworkX.

b) [3 pts.] Write a function that reads the file "new_edges.txt" and for each author, keeps track of the new collaborations this user formed during 2017-2018.

In 2017 and 2018, there were 1,757 new edges formed between existing authors. For the next tasks, pick (and recommend new collaborations for) those authors that formed at least 10 new connections between 2017-2018. In the remaining, when we talk about author X, we refer to one of those authors.

c) [5 pts.] **Recommend by number of common friends**
if non-friend Y is your friend's friend, then maybe Y should be your friend too. If person Y is the friend of many of your friends, then Y is an even better recommendation.

Write a function $common\_friends\_number(G, X)$ that given G and an author X, returns a list of recommendations for X. The authors in this list are sorted by the number of common neighbors they have with X (and are not of course already friends with X). If there are ties, you can break them arbitrarily.

d) [5 pts.] **Make recommendations using Jaccard's Index**
If $\Gamma(X)$ is the set of neighbors of X, then the metric we used in part (c), assigns to a non-friend y, the following recommendation score (with respect to X): $score(y) = |\Gamma(X) \cap \Gamma(y)|$. Jaccard's Index scales this score by taking into account the union of X and Y's neighbors. Intuitively, X and Y are more similar, if what they have in common is as close as possible to what they have together.

Write a function $jaccard\_index(G, X)$ that given G and an author X, returns a list of recommendations for X. The authors in this list are sorted by the number of their Jaccard Index with respect to X (and are not of course already friends with X). If there are ties, you can break them arbitrarily.
Jaccard Index $= \frac{|\Gamma(X) \cap \Gamma(y)|}{|\Gamma(X) \cup \Gamma(y)|}$

e) [5 pts.] **Make recommendations using Adamic/Adar Index**
For part (c), we made recommendations using common neighbors. However, when assigning a score to Y, instead of just taking a count of the number of common neighbors, we take a weighted sum of them, where the weight of each common neighbor of X and Y, call her Z, is the inverse of the logarithm of the number of Z's neighbors. In that way, we value more common neighbors that are more selective.

Write a function $adamic\_adar\_index(G, X)$ that given G and an author X, returns a list of recommendations for X. The authors in this list are sorted by the number of their Adamic/Adar Index with respect to X (and are not of course already friends with X). If there are ties, you can break them arbitrarily.
Adamic/Adar Index (y)= $\sum_{Z \in \Gamma(X) \cap \Gamma(y)} \frac{1}{log|\Gamma(Z)|}$

f) [4 pts.] **How good are the recommendations we make?**
Previously, you implemented 3 functions, that given a user X provide recommendations for this user. In this task, you will check how good these recommendations are using the actual new connections formed during 2017-2018.
You will use two different ways, to calculate the efficiency of every approach:

- For each user X, take the 10 first recommendations for this user, and calculate the number of them that were actually formed during 2017-2018. You should report the average among users X.
- For each newly formed collaboration of user X, calculate the rank of this collaboration (the index where this new node Y appears in the recommendations list for X). Report the average among newly formed edges.

e) [**Bonus Question**] [2 pts.]
Doing some literature search, suggest your own algorithm for recommend-

ing new links to a user X. Argue about the choice you make, why it makes sense to suggest users that way? How is the efficiency of this algorithm, compared to the ones you implemented in parts (c), (d) and (e)?