FACEBOOK FRIEND RECOMMENDATION SYSTEM

(Using NetworkX)

Project Report:

Abhignya Goje (700703549)
Sindhu Konatham
(7007032470)

Table of Contents

Abstract

A social network is composed by communities of individuals or organizations that are connected by a common interest. Online social networking sites like Twitter, Facebook and Orkut are among the most visited sites in the Internet. Presently, there is a great interest in trying to understand the complexities of this type of network from both theoretical and applied point of view. The understanding of these social network graphs is important to improve the current social network systems, and to develop new applications. Here, we propose a friend recommendation system for social network based on the topology of the network graphs that are formed from by considering the nodes as the users and the edges as the friendship between them.

Introduction

A **recommender system** or a **recommendation system** is a subclass of information filtering system that seeks to predict the "rating" or "preference" that a user would give to an item. Recommender systems have become increasingly popular in recent years and are utilized in a variety of areas including movies, music, news, books, research articles, search queries, social tags, and products in general. There are also recommender systems for experts, collaborators, jokes, restaurants, garments, financial services, life insurance, online dating, and Twitter pages. Facebook suggests people you could be friends with. The actual algorithms used by these companies are closely guarded trade secrets. There are two general approaches: collaborative filtering and content-based filtering.

- **Collaborative filtering** says that, if your past behavior/preferences were similar to some other users, then your future behavior may be as well. As a concrete example, suppose that you like John, Paul, and George, and other people like John, Paul, George, and Ringo. Then it stands to reason that you will like Ringo as well, even if you had never previously heard of him. The recommender system does not have to understand anything about what “John”, “Paul”, “George”, and “Ringo” are — they could even be brands of toilet paper, and the algorithm would work identically.
- **Content-based filtering** considers the characteristics of the things you like, and it recommends similar sorts of things. For instance, if you like “Billie Jean”, “Crazy Train”, and “Don't Stop the Music”, then you might like other songs in the key of F-sharp minor, such as Rachmaninoff's “Piano Concerto No. 1”, even if no one else has ever had that particular set of favorite songs before.

Dataset

In each line of the file containing this dataset, there are two integer values and these integer values are separated by the horizontal tab character (\t). Here, each integer value represents the IDs of the users. Hence, each line has the format given below, where user1 represents the ID of the first user, user2 represents the ID of the second user and \t is the tab character: user1\tuser2

Facebook has an undirected relationship format which means that when you become a friend of a user, then this user also becomes your friend as well. For instance, if you see the line given below in the file (integers are separated by a tab), it means that the user with ID 10 is a friend of the user with ID 158, and it also means that the user with ID 158 is a friend of the user with ID 10. In other words, such a line means that the users with IDs 158 and 10 are friends with each other.

Objective of the project

In this project, we implemented a collaborative filtering recommendation system for suggesting friends on Facebook. We implemented two mechanisms for recommending a new friend in a social network. For user X, we listed some *non-friends* in order, starting with the best friend recommendation and ending with the worst. A non-friend is a user who is not X and is not a friend of X. Depending on the recommendation algorithm, the list may include all non-friends or some of them.

Further, the recommendations might not be symmetric. We implemented a code in python that produces friend recommendations for U, in order from best to worst. This is done by assigning each potential friend a number called a score, where higher scores indicate a better match. Then the list is returned according to the score. Given user X, if two people Y and Z would be equally good as new friends for X (they have the same score), then they are listed in alphabetical order (for names) or numerical order (for numerical user IDs).

Innovation component in the project

In this project, the uniqueness is the calculation of lonely nodes. Lonely nodes are the nodes in the social network who are not at all the best recommendations to anyone in the social network. The clustering coefficient is an important factor which determines the loneliness of a node in a social network. The clustering coefficient is the measure of how much the friends of a user are connected to each other. The implementation says that the lonely nodes are the nodes with least clustering coefficients. The results made from this observation is that a person is not lonely because he has no or less friends but due to the reason that his friends are not friends among themselves. The discussion follows that the nodes with low clustering coefficient and are not at all recommended by the recommendation system have more suicidal tendency.

The results also show that every other friend of a user has an influence in the life of user and the next friend of the user is largely dependent on current friends of the user. The recommendation results for influence method show that the every new friendship is influenced by the current friendships.

Work done and implementation

Methodology adapted:

Software Requirements:

Spyder (Python 3.6)

NetworkX (Python

Library)

Hardware Requirements:

Any computer with python installed would work.

Methodology:

The methodology of this project consists of two core methods of recommendation. The first one is the *recommend_by_common_friends* and *recommend_by_influence.*

Recommend by 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. The best friend recommendation is the person with whom you have the largest number of mutual friends.

Recommend by influence:

Consider the following hypothetical situation.
Since Y is highly selective in terms of friendship, and is a friend of X, X is likely to have a lot in common with Y's other friend. On the other hand, Z is indiscriminate and there is little reason to believe that X should be friendly with any one of Z's other friends. Incorporate the above idea into your friend recommendation algorithm. We call this technique "influene scoring".

Suppose that user1 and user2 have three friends in common: f1, f2, and f3. In this problem, the score for user2 as a friend of user1 is

1/numfriends(f1) + 1/numfriends(f2) + 1/numfriends(f3)

where numfriends(f) is the number of friends that f has. In other words, each friend *F* of user1 has a total influence score of 1 to contribute and divides it equally among all of *F*'s friends.

Evaluation:

The following algorithm is performed 100 times on the dataset and average is calculated of all average values returned to evaluate the results

1. Two nodes are chosen at random.
2. Their friendship is removed from the graph.
3. Friend recommendations for F1 and F2 are computed.
4. Rank of F1 in F2's list of recommended friends is calculated. Rank of F2 in F1's list of recommended friends is calculated. Average of both ranks is computed.
5. Friendship is put back to the graph.

For a perfect recommendation system, the first recommendation for F1 would be F2, and the first recommendation for F2 would be F1. In general, the closer to the front of the list these recommendations are, the better the recommendation system. For a good recommendation system, the average rank should be small.

Dataset used :

Dataset is taken from SNAP (Stanford Network Analysis Project)

Source (Citation): J. McAuley and J. Leskovec. Learning to Discover Social Circles in Ego Networks. NIPS, 2012.

This dataset consists of 'circles' from Facebook. Facebook data was collected from survey participants using this Facebook app. The dataset includes node features (profiles), circles, and ego networks. Facebook data has been anonymized by replacing the Facebook-internal ids for each user with a new value.

Tools Used

Spyder (Python 3.4) is used to write the code. NetworkX library of python is used to analyze and work with the dataset. NetworkX is a Python package for the creation, manipulation, and study of the structure, dynamics, and functions of complex networks or graphs in general. It has the following features:

- Data structures for graphs, digraphs, and multigraphs
- Many standard graph algorithms
- Network structure and analysis measures
- Generators for classic graphs, random graphs, and synthetic networks
- Nodes can be "anything" (e.g., text, images, XML records)
- Edges can hold arbitrary data (e.g., weights, time-series)
- Well tested with over 90% code coverage
- Additional benefits from Python include fast prototyping, easy to teach, and multi-platform

Dataset Analysis Report

- Type: Graph
- Number of nodes: 4039
- Number of edges: 88234
- Average degree: 43.6910
- The Graph is connected
- The graph is not directed
- Average clustering coefficient is: 0.6055

Results and discussion

Recommendations for the user 1222:

Recommendation by Influence method:

[(107, 1.5041943556371893), (1888, 1.0776933333350598), (1352, 1.0612043383752092), (1377, 1.0363770157940952), (1730, 1.0127165834130822), (1663, 1.0070598176447842), (1551, 0.9766386140683303), (1813, 0.9717048924955745), (1768, 0.9555284637663647), (1199, 0.9479967116267918)]

Recommendation by number of Common Friends method:

[(1376, 97), (1833, 91), (1746, 88), (993, 86), (1390, 86), (1391, 83), (1714, 83), (1059, 81), (1516, 81), (1612, 81)]

Average Rank of Method 1 - Recommendation by Number of Common friends:

12.0

Average rank of Method 2 - Recommendation by influence:

0

Number of Same recommendations from both methods:

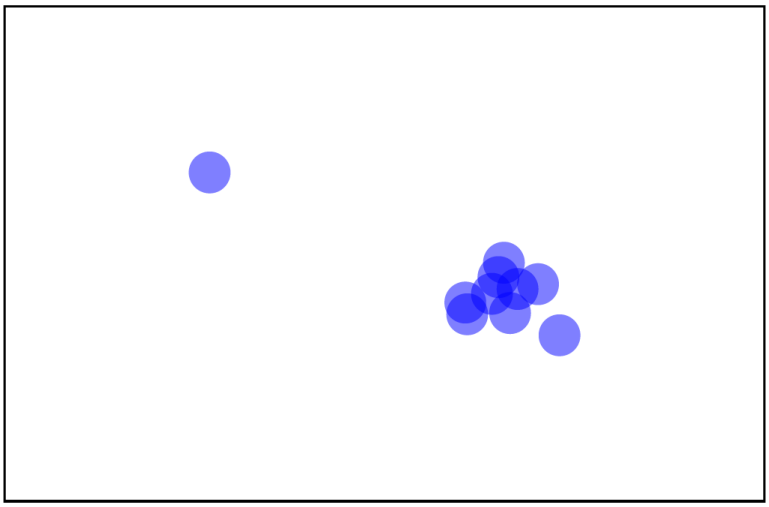
108

Number of different recommendations from both methods:

3931

Lonely Nodes:

{1431: 0.43345786633457867, 2464: 0.7420865239233283, 917: 0.22595419847328244, 1746: 0.4682241623177096, 2340: 0.7581397733681997, 483: 0.16035975518734139}



Conclusion

By observing the results, we can conclude that the method of recommendation by influence is better and more efficient than method of recommendation by number of common friends. This reference is drawn due to the values of average rank values of both methods. The average rank of influence method is smaller than average rank of common friend method. Also, the nodes with least clustering coefficient are called lonely nodes and have more tendency to be lonely and have suicidal tendency. We also say that the next possible friend is dependent on the influence score of current friends.

References

[1] https://en.wikipedia.org/wiki/Recommender_system

[2] https://networkx.github.io/documentations

[3] https://snap.stanford.edu/data/egonets-Facebook.html

[4] Jeff Naruchitparames, Mehmet Hadi Gunes and Sushil Louis (2013) “Friend Recommendation in Social Networks using Genetic Algorithms and Network Topology”

[5] https://courses.cs.washington.edu/courses/cse140/12su/homework/hw4/homework4.html

[6] https://www.python-course.eu/graphs_python.php