Social Network Analysis Community Detection

Wajih Arfaoui

3/2/2022

Contents

1. Introduction

In this assignement, I am going to undertake community detection in the **R** package `igraph`, to attempt to detect the most important community within the follower network of IESEG Twitter account. This will be implemented using one popular community detection algorithm: Walktrap.

2. Community Detection Importance

In a general note, it may be necessary to look for communities when analysing networks. Social media algorithms can utilize community detection techniques to find people who **share common interests** and keep them connected. In machine learning, community detection can be used to find groups with similar attributes and extract groups for a variety of reasons, such as finding out **manipulative groups** in a social networks.

3. Community detection

Graph theory's concept of community detection differs from clustering since the latter is considered as broader field of unsupervised machine learning that deals with multiple attribute types to group similar data points while the former is mainly applied for network analysis and depends on single attribute which is edges.

The community detection methods, aim to locate dense subgraphs inside directed and undirected graphs by optimizing various criteria and in most case using heuristics.

The method I am going to focus on in this project is:

- **Walktrap Community Detection**: This approach was first developed by Pascal Pons, and it is an algorithm in graph theory, used to identify communitites in networks based on random walks that are used to compute distances between nodes. This approach's basic intuition says that random walks on networks tend to get trapped into highly connected parts corresponding to communities. So, Walktrap uses the results of random walks to assign nodes into groups with small intra and larger inter-community distances via bottom-up hierarchical clustering.

4. Dataset & preprocessing

As mentioned before, I am going to to analyze **IESEG** Twitter account network. The objective in this analysis is: Visualize the top communities from IESEG account, find out which account has the potential to spread information widely, Calculate the centrality, and find out who is the key player in IESEG network.

First step, is to scrap @IESEG account data and its followers using the rest API from developer account. Considering the retrieval limitation of this method, I will be applying filtering criteria on the followers, so we end up getting data only about users respecting these ones:

- `Followers_count` > 100, `following_count` > 100, `favourites_count` > 10, and created a `new tweet` at least 2 months ago.

After gathering active users, I will move to collect their followers. Knowing that Twitter API has retrieval limit, I will have only half of the followers. The list I receive from this loop, will be binded to dataframe then I will convert the username to `user_id` by left joining with active followers data and cleaning **NA**s.

The same thing will be applied to gather IESEG's followers' followers, and since friends_count is way more higher than followers_count, I will be minimizing it by retrieving 40%. Then, I get the active accounts' user ids by using left join.

Now that I have both following and follower data, I will build `mutual` dataframe to make sure the network is a strong **two-side-connection network**. Mutual will reference accounts who follow each other. To find that, I need to split `ieseg_friend` data by every unique `active_user`, then we find every account in the following column that also appears in `ieseg_follower$follower`. The presence in both columns indicates the user is following each other.

5. Social Network Analysis

I will start by building two separate dataframes;`nodes` dataframe from every unique account in `ieseg_mutual`, and `edges` dataframe that contains pair of accounts. After that, I will create graph dataframe using `graph_from_data_frame` function from `igraph` package.

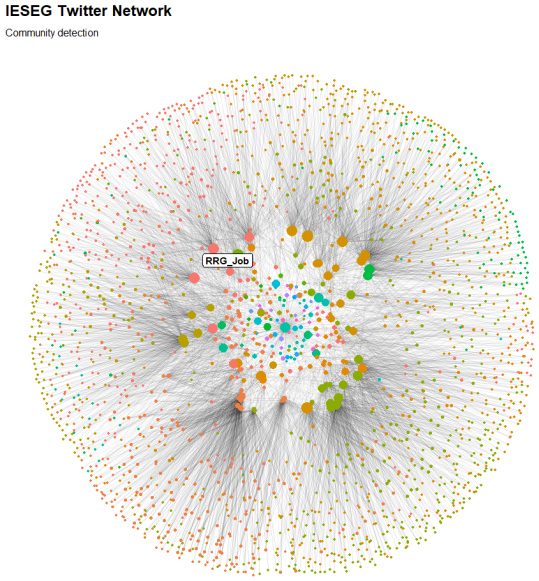
As you can see below, I use the graph dataframe `tbl` to build **communities** and calculate **centrality** metrics as follow:
- Community detection using `walktrap.community()`
- Degree of centrality using `centrality_degree()`
- Betweenness centrality using `centrality_betweenness()`
- Closeness centrality using `centrality_closeness()`
- Eigen centrality using `centrality_eigen()`

	name	community	degree_c	betweenness_c	closeness_c	eigen
1	3163920275	3	1	0.000000e+00	0.2520854	0.002655159
2	1444119200210067457	3	1	0.000000e+00	0.2520854	0.002655159
3	1456384500091564034	3	1	0.000000e+00	0.2520854	0.002655159
4	4894484579	3	1	0.000000e+00	0.2520854	0.002655159
5	1429998774156419076	3	1	0.000000e+00	0.2520854	0.002655159

Now that, I have scores for different metrics, I will try to find to find top accounts in each centrality, in order to detect who is the most **influential user** in the IESEG network based on centrality.
From the table above, account *"700418206138245120"* appears in the top 6 of all the centrality metrics. This account has the most degree in the network, and it is surrounded by important accounts since it has a high eigenvector. Hence, we can consider this account as the most influential account of IESEG Twitter network. To know to which person/entity this ID number refers, I will use the `user_lookup()` from the `rtweet` package to find that the most influential account belongs to **RRG__Job** which is an official Twitter account for **Renault Retail Group* specialized in promoting jobs' offers.

	degree	betweenness	closeness	eigen
1	7004182061382451 20	IESEG	IESEG	IESEG
2	773340859303682049	7004182061382451 20	7004182061382451 20	193677301
3	1963164662	773340859303682049	193677301	8062702
4	8062702	1963164662	1963164662	780292093839220736
5	564806527	14464283908405370 90	564806527	773340859303682049
6	193677301	564806527	110362528	7004182061382451 20

Finally, I will plot my network. To do so, I'll scale the nodes by degree of centrality, and color it by community, but since our network is too large, and it won't make sense to plot all of it, I choose to filter the network by just visualizing top 1000 based on the degree of centrality to get this result.



Through this graph we can see that the "random walk" community detection algorithm is approximately having the same structure as *stress* layout algorithm. IESEG appears in the middle connecting all the communities together while showing the user label of the account with the highest degree,betweeness and closeness values, which is the **RRG_Job** account. Considering this fact, we can say that the red community, to which this user belongs can be considered as the most important one for IESEG, because it has the potenatial to soread the information widely and fast.

6. References

https://www.rdocumentation.org/packages/intergraph/versions/2.0-2
https://tidygraph.data-imaginist.com/reference/group_graph.html
https://github.com/western11/Social-Network-Analysis-Twitter-Network
https://medium.com/analytics-vidhya/social-network-analysis-in-r-38fbf2512290
https://cran.r-project.org/web/packages/graphlayouts/readme/README.html
https://towardsdatascience.com/how-to-model-a-social-network-with-r-878b3a76c5a1
https://search.r-project.org/CRAN/refmans/igraph/html/communities.html