CSE 472
Social media Mining
Project – 1
“Friendship network of cricketers in twitter”

Done by
Trilok Tourani (ttourani@asu.edu)
Nagarjuna Battula (nbattul1@asu.edu)

Table of Contents

1. Introduction

Twitter is a very popular website containing millions of active users and a lot of information that is shared every day. Twitter's concept is simple. There are a lot of users and they can share information with each other. In addition to it, the users can follow each other in twitter. So the users using twitter get the feed from the other users they follow. For example, if user A follows only two users, say user B and user C, user A gets all of his/her feed from user B and user C. In this way, the information between users is shared and spread across twitter.

Also, if two persons follow each other, it is obvious that the persons are interested in each other's information. In twitter, we call them as friends. These two persons get each other's tweets, retweets and likes.

In this project, we have created a friendship network between 6 cricketers and their followers. This project is divided into three steps:

Step 1: Crawling the data

Access twitter api and get the profile ids of 6 cricketers. And then extract 40 followers' ids from each cricketer randomly. All these people including cricketers are treated as nodes. This network requires to be a directed graph. So, if a person A follows person B, then a directed edge is drawn between A's node and B's node. Accordingly, we have created an adjacency matrix for all the nodes that are in the network. This network graph consists of 217 nodes.

Step 2: Creating the network graph and visualizing it

As we have an adjacency matrix and the nodes, we create a graph based on it. We have used networkx library in python to create the graph and the matplotlib library to visualize it.

Step 3: Calculating and visualizing the measures of the network

Now that there is a graph, we can use it to calculate some standard measures associated with it. We have calculated degree distribution, eigenvector centrality and pagerank centrality for the graph using the networkx library and also visualized these measures to understand better as there are a lot of nodes associated with the network.

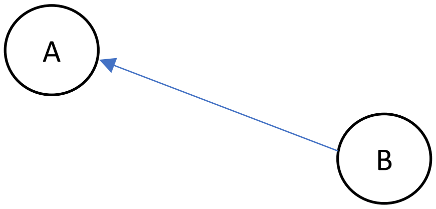

Figure 1 – Sample follower graph

2. Step 1 - Crawling the data from twitter

2.1 Getting access to twitter API:

To do this, one should have a twitter account. Then, we need to go to https://developer.twitter.com/ to apply for a developer account in twitter. As we have done this project for academic purposes, we have applied for the student twitter developer account giving appropriate details like phone number, email id and the description of the project. The twitter review community handles this application request and gives a twitter student developer account for free.

After this, we need to create an app in the developer account by giving the appropriate details like app name and description of the app. After the app is created, the keys and tokens related to the application are generated. These keys are used for the authorization of the usage of twitter developer API.

2.2 Crawling the data:

Our project aim is to build a friendship network of 6 cricketers and their followers each. So we need to extract some of the followers of each cricketer. The cricket players used in this network are:

1. Virat Kohli (@imVkohli)
2. Sachin Tendulkar (@sachin_rt)
3. Mahendra Singh Dhoni (@msdhoni)
4. Chris Gayle (@henrygayle)
5. Rohit Sharma (@ImRo45)
6. Brian Lara (@BrianLara)

We used *twython* library to use the twitter api in python. "*twython*" is an open source library that uses twitter api to provide the twitter api functionalities to the twitter developers in python.

Using the twitter keys and tokens, we have authorized the twitter api connection in python by running *Twython(consumer_key, consumer_secret, access_token, access_token_secret)* function. This function shouldn't return any error if the keys and tokens that are provided are true. The output of this function is stored in a variable (say twitter_user)

After this, we get 40 followers from each of the cricketer using the function get_follower_ids as follows

*twitter_user.**get_followers_ids**(screen_name=player, count=40)*

This function needs to be run in a loop to get the followers of all the cricketers.

Note that some of the followers that are extracted are common followers to two or more cricketers. So we need to make a set of the followers so that there is no repetition of the nodes.

Using the *get_friends_ids(id=user)['ids']* function, we can get the friends list of all the nodes. We compared the friends list obtained in this step with the nodes that are obtained in previous step and created an adjacency matrix for the nodes to list out the friends.

The code runs in python installed systems using this command in terminal or command prompt:

```
python filename.py
```

3. Step 2 – Creating the network graph and visualizing it

After crawling the data in step 1 in the form of adjacency matrix, we need to create a graph for the matrix and visualize the graph.

In this project, we have used the **gephi** tool to create the graph for it.

Sequence of steps for creating the graph in gephi:

Create a new project in gephi -> open data laboratory -> select edges -> import spreadsheet.

Select the csv file containing the adjacency matrix and import the edges. This will also create the graph. To view the graph, click overview.

Graph figure

We have created a directed graph and visualized it by ranking nodes with respect to their in-degree.

Observations

1. All cricketers have the highest in-degree as we are targeting their followers.

2. We can see clusters forming for every cricketer with Brian Lara, and ImRo45 having comparatively less followers in this case.

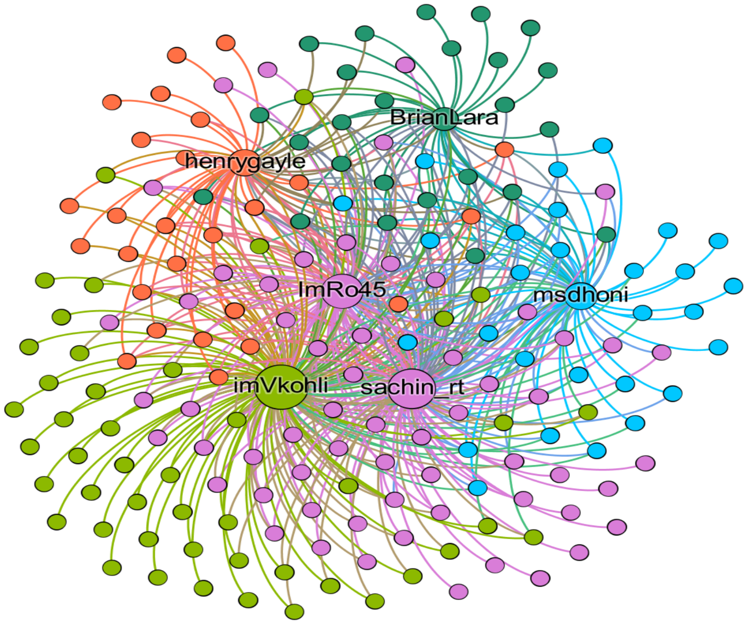

Figure 2 – Friendship network graph

4. Step 3 - Calculating and visualizing the measures of the network

4.1 Degree Distribution

We can see from the degree distribution histogram that :

1. The highest degrees are the ones on the right of 6 cricketers having the most popularity.

2. There are some followers who follow multiple cricketers or each other (even though we took random 40 followers out of every cricketers list)

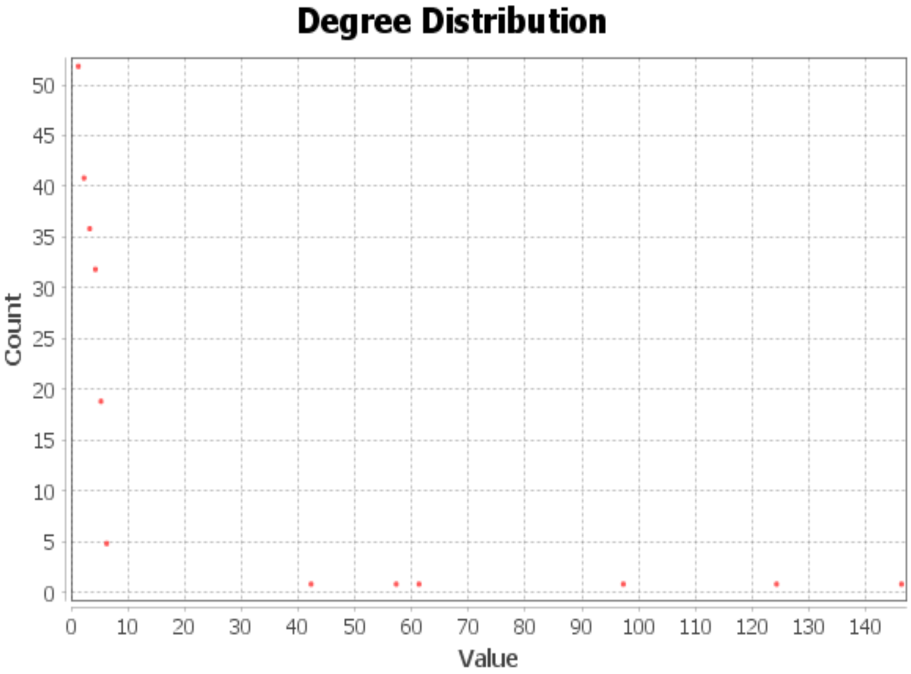

Figure 3 – Degree distribution

4.2 Closeness Centrality

More central a node is, the closer it is to all other nodes. (Ranging from 0.352 to 1 in our case)

1. Some cricketers don't have enough closeness centrality compared to some of their followers. This is because these followers are connected to each one of the cricketer and some follower nodes.

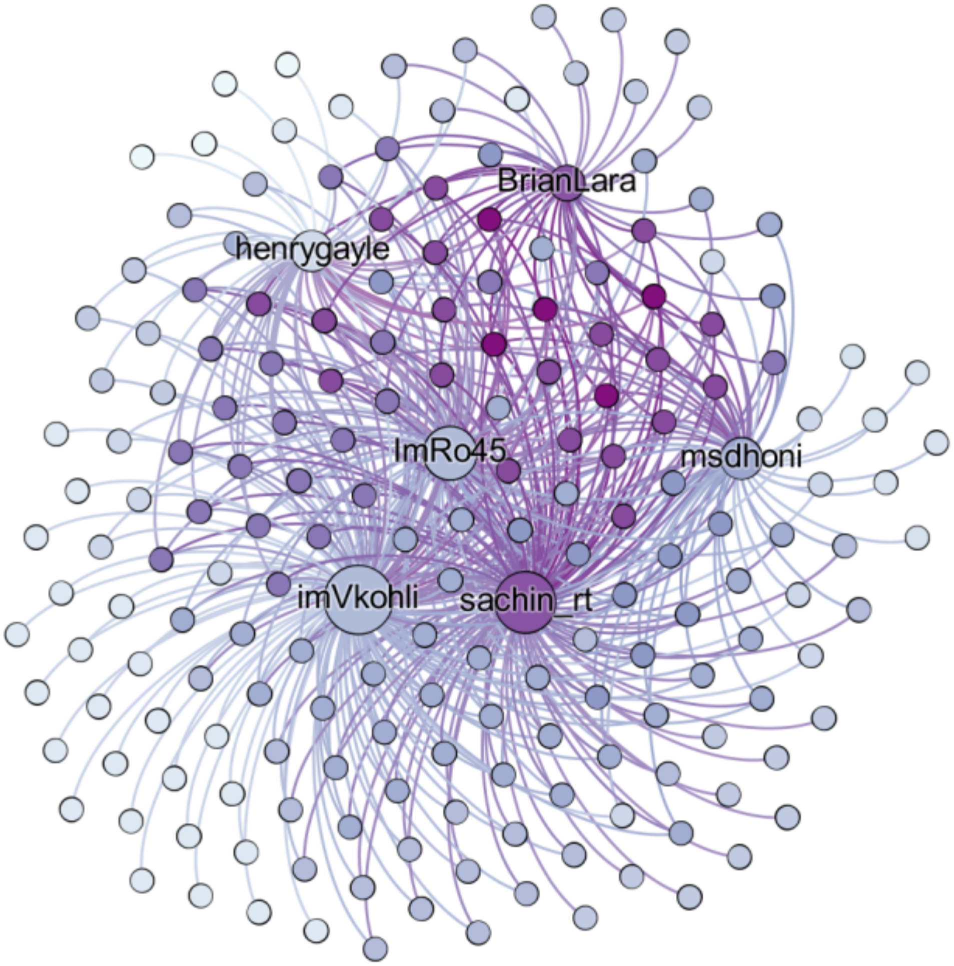

Figure 4 – Closeness Centrality

4.3 Betweenness Centrality

Measure of centrality based on shortest paths. (Ranging from 0 to 384 in our case)

1. Sachin has the highest centrality as he has many mutual followers with others.

2. All the followers have 0 centrality as they don't follow anyone else other than the cricketers, so the cricketers are the only ones that come in every shortest path.

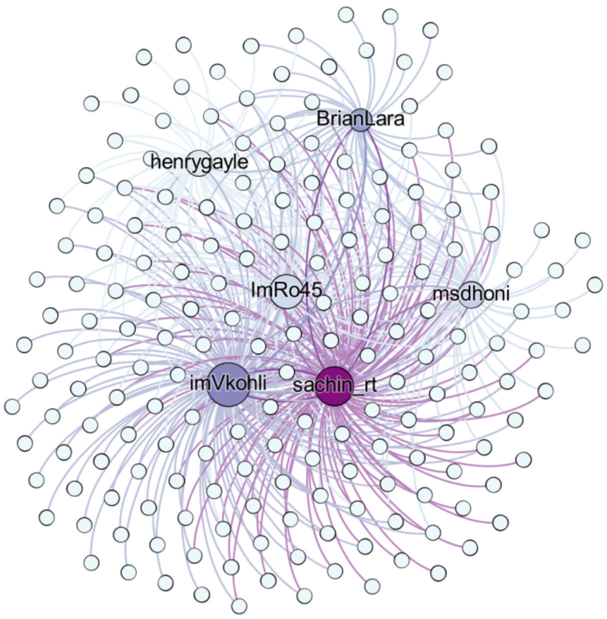

Figure 5 – Betweenness graph

4.4 Clustering Coefficient

Measure of the degree to which nodes tend to cluster together. (Ranging from 0 to 1 in our case)

1. All cricketers are into one cluster based on the data. (with less difference between their coefficients)

2. All the followers which follow only one are mostly in the one cluster (pink)

3. All the followers which follow more than 1 are in another cluster. (green)

The clustering can be seen in figure 6.

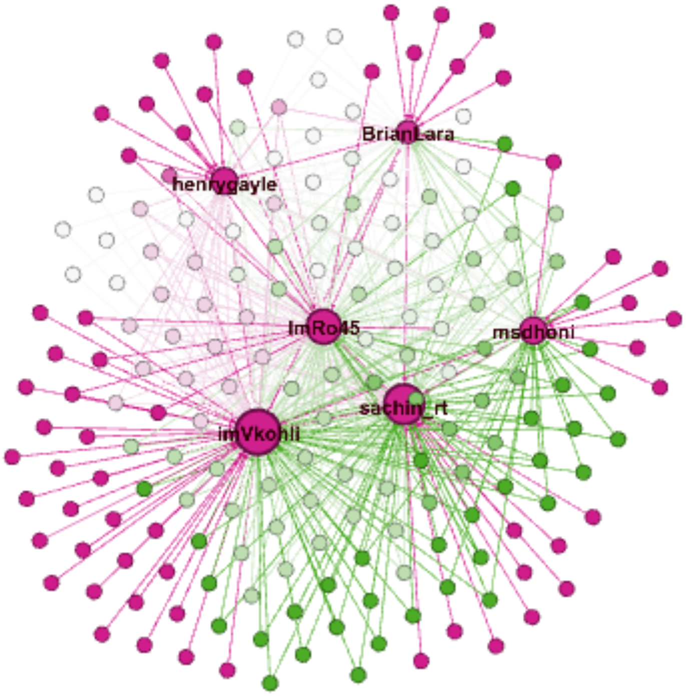

Figure 6 – Clustering Coefficient