DS210 Project: An Analysis of the Six-Degree Problem

For my DS210 project, the specific problem I would like to investigate is "Six Degrees of Separation," which theorizes that all people are six or fewer social connections away from each other due to a chain of "friend of a friend". I aim to assess this hypothesis's validity in general and explore its applicability across various demographic groups.

The specific dataset that I would like to analyze is sampled from the Facebook network. Facebook is a significant social media and networking platform operated by the tech giant Meta. Established in 2004 by Mark Zuckerberg and gaining popularity primarily among university students in the United States, it has evolved into one of the world's most extensive online social networks, boasting around 3 billion users[0]. The structure of the dataset, representing real-world social connections, offers a rich ground for graph analysis and social network patterns. The dataset can be accessed via the following link[1], part of the Stanford Network Analysis project. The dataset preserves the structure of the undirected network with a list of paired connected edges between two nodes. Anonymized information for each node, namely a user, is available but stored in a format I will explain later. I will start with analyzing the whole network and then use the same framework to analyze users with different demographics. The flow in my write-up is also the flow for the code implemented in my main function.

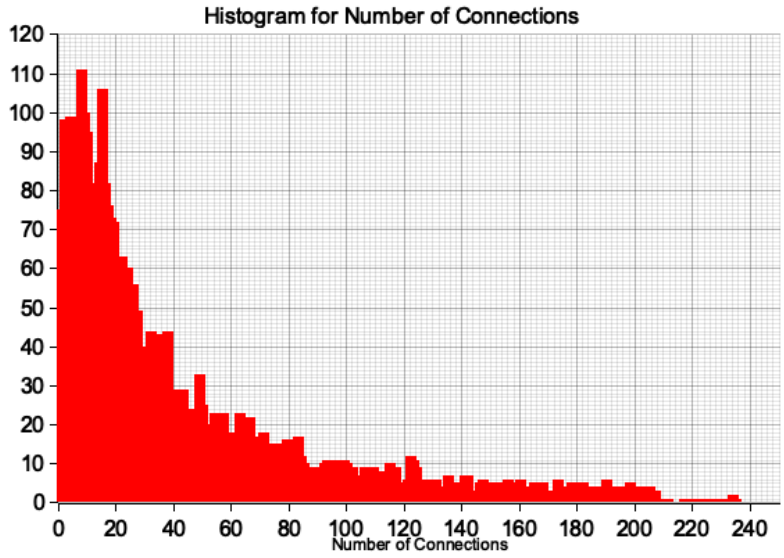
I start by reading the list of edges for the whole network stored in the txt file and use a vector to keep the list of edges from the original txt file and a HashSet to store the index of all nodes to evaluate the total number of nodes in the network. I started with some exploratory data analysis on the graph. According to my code, the maximum node index is 4039, and there are 4039 nodes in the network where index and size are matched. 88234 undirected edges connect these nodes, which is a reasonably complex network.

After that, I utilize the edge list and the calculated network size to initialize a custom-defined struct named 'Graph' to represent the network. This is also the primary entity being processed in my code. Such initialization also translates the edge list into an adjacency list for more efficient processing. Utilizing the adjacency list, I can examine the number of followers or friends each user has and analyze their distribution. This is the code output.

| Statistics | Number of friends each user has |
|---|---|
| Mean | 43.69 |
| Median | 25 |
| Maximum | 1045 |

| | |
|---|---|
| Minimum | 1 |
| standard deviation | 52.41 |

As shown from the table, with a mean larger than the median, the distribution for the number of friends can be right skewed. The variable has a standard deviation of around 52, which is relatively dispersed. People with the maximum number of friends connect almost one-fourth of the graph. A histogram is given to show the distribution. The number of connections may follow the power law distribution.

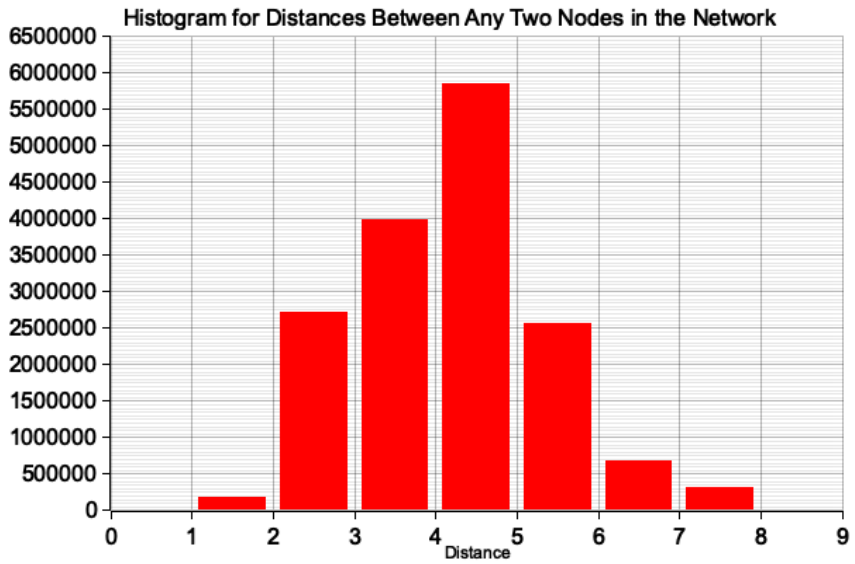



As follows, I would like to examine the distance between two nodes in the network to examine the Six Degrees of Separation hypothesis. To do that, I will have to calculate the distance between any two nodes in the network on an exhaustive basis, which will yield a matrix of 4039*4039 to store the distances. To begin with, I utilize the Breadth First Search algorithm to determine the number of connected components in the network. The algorithm starts with a given node and explores all nodes at the present depth before moving on to the nodes at the next depth level. Please note that part of the implementation is referenced from [2]. According to the output of my code, fortunately, my network has only one connected component. As follows, I use the Breadth First Search algorithm again to find the paired distances between every two nodes in the network. The algorithm is placed inside a for loop to loop over every index of the network. The results are shown as follows.

| Statistics | Distance between two nodes |
|---|---|

| | |
|---|---|
| Mean | 3.69 |
| Median | 4.00 |
| Maximum | 8.00 |
| Minimum | 0.00 |
| standard deviation | 1.20 |

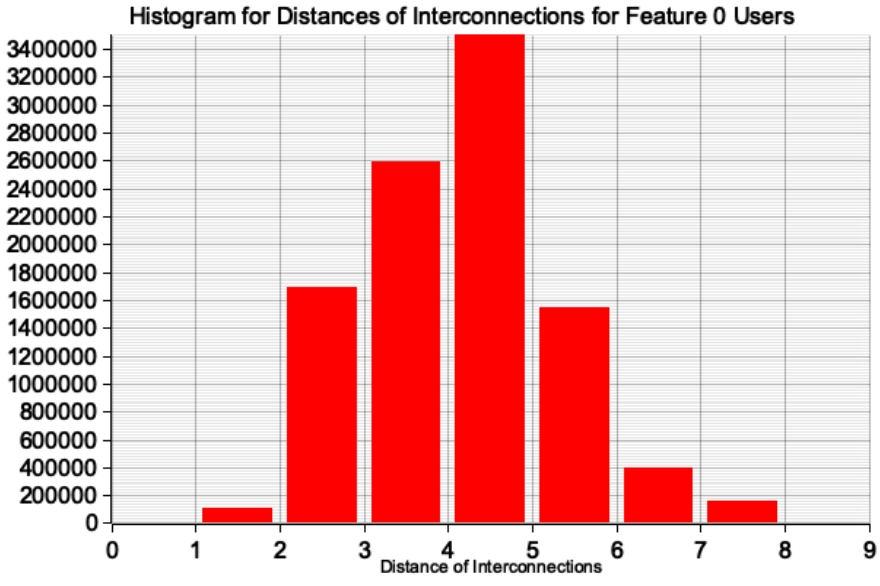
The median is similar to the mean, suggesting a possible symmetric distribution. The maximum distance between two nodes is 8. The standard deviation is 1.2, which is not too dispersed, meaning that distances between two random nodes are relatively consistent. A histogram is given to visualize the distribution. As shown from the plot, we can see that such distances could follow a Poisson distribution or even a normal distribution. Most of the time, it only took 2 to 5 steps to connect two users with a chain of friends, but it can take 8 steps to connect two users in my network. Based on the results, the Six Degrees of Separation hypothesis may not fully apply to our network, and it may be even less applicable in the broader context of the entire Facebook network.



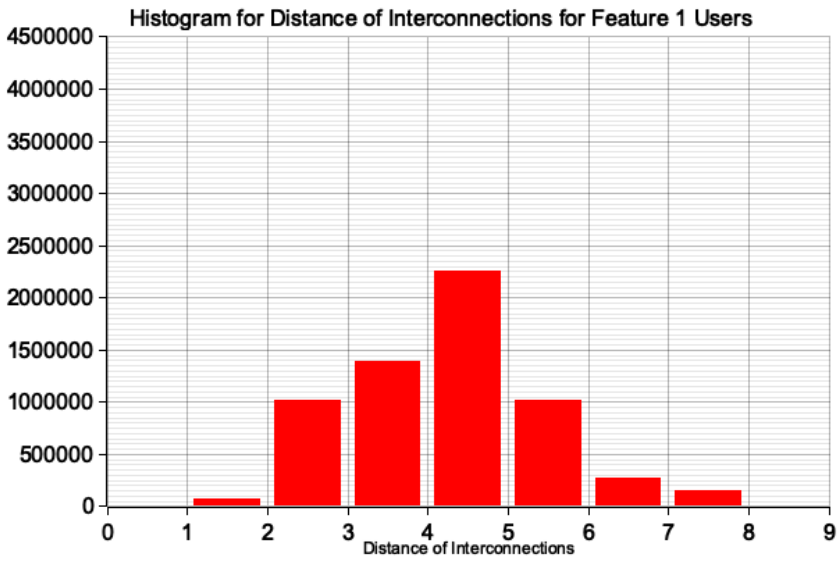

Next, we analyze if the hypothesis still holds for different demographics. The source data of the network comes with certain anonymized features. However, these features are binary and are stored separately in a file that partitions the total network into 10 ego networks surrounding specific users. The ego network can be connected back to the original network. However, the features for each sub-network could be different, and the same feature can also be stored at a different index for each sub-network. So, for each ego network, there is a file ending with

".featnames" to store the metadata of the features it has, a file ending with "feat" that stores the features for all nodes in that ego network, and a file ending with "egofeat" that stores the features of the ego node itself. So I wrote a function to loop over all these ego networks, for each iteration, it will access the metadata of that ego network to get the index of the feature of interest and then access the "egofeat" and "feat" file to get the feature value. It utilizes a HashMap to store and keep track of index-feature value pairs across 10 sub-networks. If a specific feature is missed for a particular network, it will skip that network and continue the loop to get all possible data. With a specific feature, I am able to partition the above distance matrix of 4039*4039 according to their index and redo the six-degree analysis. This can tell us the distance between users with specific demographics and all other nodes in the network. For demonstration purposes, I utilize feature 77, which is anonymous gender. This is the output of my program.

| Statistics | Distance between users with gender A to other nodes in the network |
|---|---|
| Mean | 3.65 |
| Median | 4.00 |
| Maximum | 8.00 |
| Minimum | 0.00 |
| standard deviation | 1.17 |



| Statistics | Distance between users with gender B to other nodes in the network |
|---|---|
| Mean | 3.75 |
| Median | 4.00 |
| Maximum | 8.00 |
| Minimum | 0.00 |
| standard deviation | 1.23 |




According to the output, there is no statistically significant difference between users with gender A and gender B in terms of their distances to other users in the network. As shown from the plot, the distributions are pretty similar to the total network but are more dispersed and normally distributed for users with gender B. So, the six-degree separation also does not hold for different genders alone.

To reproduce my results, type cargo run - - 77 for my final project. The code allows for segmentation and comparison between all other features. However, you might have to adjust the plot for better visualization. The feature number of interest must be given for the code to run, and you can look at the metadata to make your own choice. Even if part of the feature data is missing, the comparison between different users can still run on available data.

Generally speaking, I learned from the data that the number of connections between users follows a power-law distribution, and the distance between any two users in my data follows a Poisson or normal distribution. Such distances do not have apparent differences between genders, and the Six Degree of Separation does not hold in my network.

## References

[0] https://en.wikipedia.org/wiki/Facebook
[1] https://snap.stanford.edu/data/egonets-Facebook.html
[2] Sides from DS210 Lecture 27/28 by Professor Leonidas Kontothanassis

## Example Output

#please note that the pictures are stored automatically in the image folder

The maximum node number is 4039

The network size is 4039

The total number of edges is 88234

The mean number of friends each user has is 43.69101262688784

The median number of friends each user has is 25

The minimum number of friends each user has is 1045

The maximum number of friends each user has is 1

The standard deviation of number of friends each user has is 52.41411556737518

The network has a total of 1 component(s)

The mean of inter-connection distance is 3.691592636562027

The median of inter-connection distance is 4

The minimum of inter-connection distance is 8

The maximum of inter-connection distance is 0

The standard deviation of inter-connection distance is 1.1953735814053639

The mean of inter-connection distance for feature0 is 3.6587791604766036

The median of inter-connection distance for feature0 is 4

The minimum of inter-connection distance for feature0 is 8

The maximum of inter-connection distance for feature0 is 0

The standard deviation of inter-connection distance for feature0 is 1.171967688994475

The mean of inter-connection distance for feature1 is 3.745064075781959

The median of inter-connection distance for feature1 is 4

The minimum of inter-connection distance for feature1 is 8

The maximum of inter-connections distance for feature1 is 0

The standard deviation of inter-connection distance for feature1 is 1.2306898710260143