Spotify Network Analysis

Carlos Cañellas, Martina Armas, Rodrigo Camarena, Nicole Vega

Social Network Analysis

November, 2020

Network layout and obtaining SNA measures

The network is built from one starting actor (Bad Bunny), hence it is an egocentric network. Our undirected network has 61 nodes and 420 edges, every node is an artist on Spotify and the edges link them.

We decided to make every relationship undirected because the information given by the API only gives us information on Martina's interactions and data. As interacting more with some artists may change the results, we are assuming that every actor is connected to the others within the network, directly or through others.

After having created the Gephi graph, we decided that in order to see better the relationship between the nodes it would be useful to color the nodes by degree and make the size according to the number of followers. Moreover, we decided to use Force Atlas 2 because of the formula of repulsion and attraction it uses.

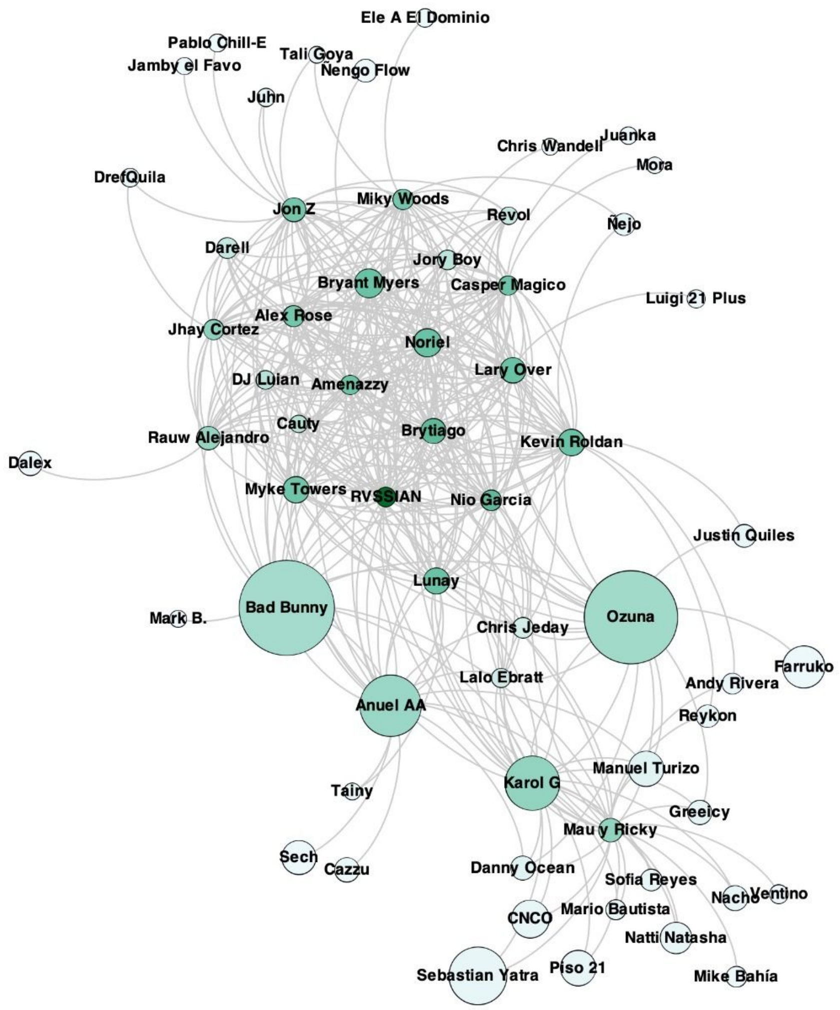


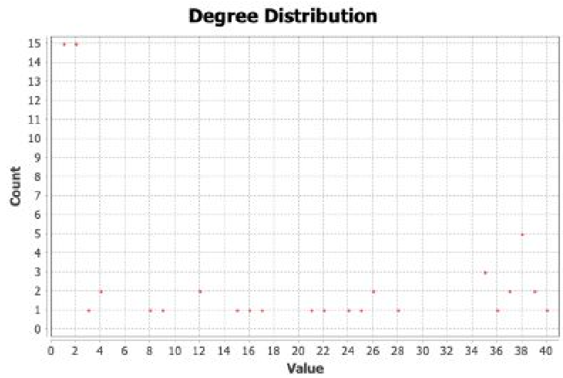
Figure 1: *Network render of Bad Bunny related artists in Spotify. The nodes represent the artists and the edges between them represent they are a related artist . The size of nodes represent the number of followers they have on Spotify and the color, the degree.*

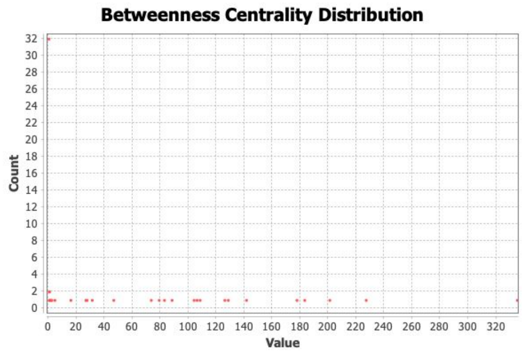
SNA Measures

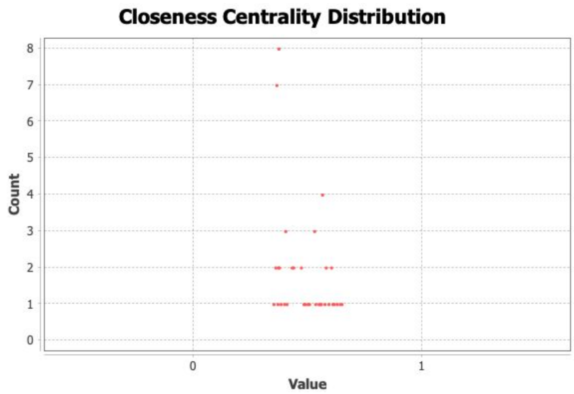
1. **Network Density**= 0.23.
2. **Average distance**= 2.26
3. **Diameter:** 4
4. **Average Degree:** 13.77

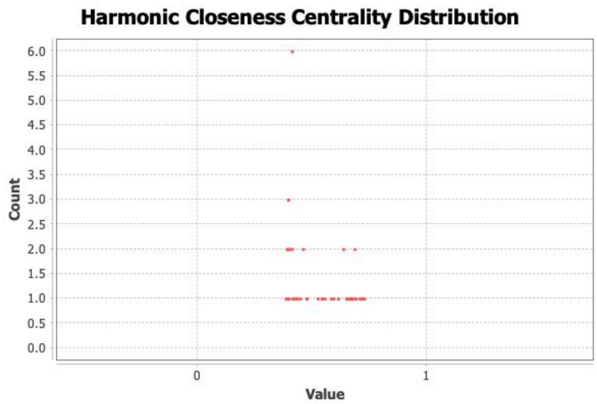
Results:

Average Degree: 13.770

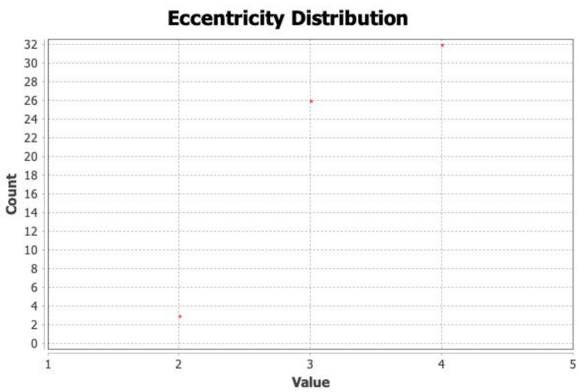


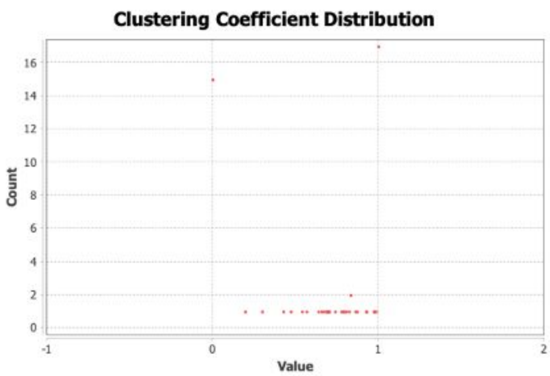






5. **Average Clustering coefficient:** 0,823





6. **Number of components:** 1

Analysis and conclusions based on the graph and measures

1. Network density= 0.23

We have a network density of 0.23. This is a low network density, which means it is poorly connected and we expected a higher one as it is an egocentric network. This can be because of the outer edges and nodes with very few connections. If we were just analyzing the cluster in the upper part, we could expect a higher network density. This is an opportunity to continue experimenting and collecting data from more than the main artists we've explored, if we do this, as we are talking about a very related genre (reggaeton), there is a high likelihood that the density of the network will increase.

2. Average distance= 2.26

When developing an egocentric network, the links typically don't go past the first order zone, which means that there is always one step from the ego (Bad Bunny) to every other node. All the geodesic distances from the ego to the other nodes would be 1 by definition. However, the case for this Spotify network varies as the network extends beyond the first-order connection, as it contains, to some degree, artists that are also connected to Bad Bunny related artists.That's why we got 2.26 for the average distance in this SNA.

3. Diameter= 4

The diameter is the length of the longest minimum network path and in this SNA is 4, which indicates that our network isn't very wide. As we have explained beforehand, even though we are in front of an egocentric network, our network presents relationships that go further than the ego (Bad Bunny) and another artist. There are edges that go beyond the first-order connection. Having Diameter 4 points out that we can find relationships of third-order which basically means that these types of connections are produced between nodes positioned at the third and fourth layer having Bad Bunny's node as our reference point. Although, diameter 4 sounds reasonable taking into account this SNA has been done only with 61 nodes. It is clear it could increment even more if we extend our network size as we are dealing with the most trending music genre of the last decade.

4. Average Degree= 13.77

The average degree is 13.77, this means on average the nodes have 13.77 connections to other actors. As we explored 21 reggaeton singers related artists, we expected this number to be high, but we weren't considering the artists in the outer edges of the graph which lower this degree as they are connected to less artists than the other nodes. In our network the darker the node, the higher the degree, thanks to this we can see that "RVSSIAN" is the most recommended artist, this can be confirmed in the edges table, where he is referred by 19 other artists.

5. Average clustering coefficient= 0.823

For this measure, we ended up with a value of 0,823. The average clustering coefficient shows how interconnected the network is between individual nodes, or another way to say it, it indicates if different smaller networks appear within the bigger network. This also indicates if the nodes interact with each other and the probability of two random nodes to be connected to each other. As this value goes from 0 to 1, in our case a value of 0,823 is a good indicative of the interconnection within our network, where it shows that most of our nodes are connected between each other.

In our case this is a normal thing, as most artists in these types of genres tend to "cluster together" to make songs, which most of them appear at least in one song as a *featured* artist. Moreover, we can visually see on the figure below a huge in-cluster between several artists in the middle, which could explain the high cluster coefficient value we ended up with. On the other hand, we can also see that there are few artists which are related only to another single artist (nodes on the outside of the graph which only have one edge). Finally, to conclude on the value of this measure, we can also visually see the interrelationships of each node based on their color, the more green, the more Spotify's related artists and radio are determined by algorithms which look at what people listen to alongside your music. So if I put your music in a playlist alongside artist X & artist Y then artists X & Y are more likely to be shown as related to you or played on radio.

There's no manual way to change these. You can start to affect it by creating a sharing playlists of your music with other music you think is more related. This will increase your streams and the bands you want to be associated with, increasing the chance of them becoming related artists.

6. Number of components= 1

Our network is composed of only one component. We expected this as we created a network of related artists starting with one specific artist (Bad Bunny). If we decide to introduce new nodes that start from a different artist in the same genre we can expect the component to remain one. On the other hand, if we decide to introduce new nodes starting with an artist from another genre we can expect to have a different component. This is because of how the Spotify algorithm works; giving us artists taking into account the rhythm of their songs, descriptions of their music and what the user hasn't heard. For instance if we start adding nodes from the Beatles, we will not have 1 component any more, but if we add them from J Balvin at some point it will connect with the current network.

Conclusions

How the Spotify algorithm works is not published on the web. People suggest it is according to the rhythm of the songs or the trending artists, however, from the analysis conducted, we came to the conclusion that Spotify's recommendation algorithm seeks for users to amplify the network of artists they normally listen to. Our information could be biased up to an extent as this network is not made randomly as it is created with data from an algorithm that suits Martina's interests. We found this interesting because if we had gathered the data from another account the artists may be different.

The measures obtained showed some insights of the structure of this network. The average degree shows us who are the musicians that are more suggested by Spotify to Martina (RVSSIAN, Brytiago, Nio Garcia, Kevin Roldan, Noriel and Lunay). Actually, we made Martina listen to these artists without her knowing that these were the most recommended to her, and we found out that she really liked them, which tells us that the Spotify algorithm knows her interests and that there is a high probability that she will like the artists that are more recommended to her.

As we have seen so far, the density result of our network, points out we are in front of a weak connected network. Even though the majority of the results conduct us to classify this network as a low connected one it is important to understand before the purpose of the algorithm and then, just only after, draw conclusions. As mentioned previously, this algorithm is intended to display the most suitable artists for Martina's tastes. This not only means displaying related artists to Bad Bunny but also taking into account several factors to filter them. The algorithm is not only seeking to offer the most related artists but also the ones that most probably are new to Martina. That is the main reason why we haven´t found J Balvin in this network. Is J Balvin related enough with Bad bunny in order to be displayed? Yes, absolutely. Nevertheless, Martina already knows J Balvin albums from a to z. It doesn't make any sense to display it on the network. Value is not added. The main intention of Spotify with this algorithm is to keep Martina's engagement as long as possible, finding new artists, enjoying new music and thus keeping her subscription. So, as the algorithm is trying to display not only suitable but also newest artists, the higher the probability to be in front of a low density network.

On the same path, our average degree result would be unexpected if we talk only about an egocentric network but then we can understand it is quite reasonable when we introduce the algorithm factor to the equation. Our average degree is closer to the diameter because the algorithm wants to go further than the first-line of relation so it can offer Martina new artists. On the other hand, in our case a high value on the diameter, the better. This is because, Bad Bunny being the starting point or the "main node", when trying to find new artists which are related to him, it can differentiate and identify many artists in order to find a non-so well known artist but which could interest Martina based on previous nodes. That is to say, the further we can go in our network's layer the higher the probability of an unknown but interesting artist is displayed to her. Hence, the low result is mainly caused because we have only studied a small portion of our network due to Spotify's restrictions when sharing results with the developers and the size of the network as a whole. This could also be improved if we counted with a bigger network with more artists, as some of our nodes are marginated because there only exists a single relationship with another single node. Furthermore, as explained previously, maybe our results could have been different if we did the analysis of the network based on another person's Spotify account. However, maybe this result may also be biased because as we get further away from the "main node", the probability of finding an artist that Martina will like can decrease. It can be a double-edge sword.

Regarding average clustering coefficient, it goes as expected. Result is quite good considering we want to display related artists that will meet Martina's tastes. So, if we can have nodes (artists) that not only relate to Bad bunny but also with the other related artists the better for the cause. The odds are incremented. So, 0,823 is quite a good clustering result. Last but not least, the number of components don´t disappoint. We obviously were expecting 1 as we are working with an egocentric network where the edges and nodes revolve around one, the ego (Bad Bunny).

At first we thought our Network could be useful for artists and managers to create collaborations and publicity with related artists, but as it is different for every user we no longer think that, the algorithm is more related to the behavior of the user inside Spotify than the real life social networks of the artists. We found this concerning, as artists or managers wouldn't have useful information from the network, but then we found that Spotify offers a service for artists that would cover this need as it provides statistical information about their music. Our recommendation would be for Spotify besides recommending less popular artists to recommend the ones that the artist the user is exploring has collaborated with as they are likely to have music the user may enjoy. At the end we realized that this network would be resourceful for Martina and her friends with similar music taste. It would be interesting to create a new network based on what new artists Martina likes and dismiss the ones she doesn't, obtaining her personal social network of reggaeton artists she actually likes and compare it to the original one.

It would also be interesting to explore the network of a rising reggaeton artist and see how far he is from the network we created. Also, we could try to see if it's possible to change the recommendations by creating new playlists and listening to less popular artists.

References:

Spotify for Developers (no date) Spotify, Retrieved from: https://developer.spotify.com/technologies/metadata-api/search/

Spotify Community (no date) "Related artists" – how do Spotify decide what artists are related to a specific artist?" Spotify. Retrieved from: https://community.spotify.com/t5/Content-Questions/quot-Related-artists-quot-how-do-Spotify-decide-what-artists-are/td-p/1185597.

Marsden, P.V. (2002), "Egocentric and Sociocentric Measures of Network Centrality", Social Networks, vol. 24, no. 4, pp. 407–422, doi:10.1016/S0378-8733(02)00016-3.

Spotify For Artists (nd) "How does Fans Also Like work?" Spotify, online. Retrieved from :https://artists.spotify.com/faq/profile#can-i-create-and-edit-my-artist-playlists-using-spotify-for- artists

Jacomy. M, Venturini, T. Heymann, S et at (June 10, 2014) "ForceAtlas2, a Continuous Graph Layout Algorithm for Handy Network Visualization Designed for the Gephi Software" Retrieved from: https://journals.plos.org/plosone/article?id=10.1371/journal.pone.0098679

BookDown. (nd). "Introduction to Ego-Centric Networks" Retrieved from: https://bookdown.org/chen/snaEd/ego-intro.html