Written Report (~2 pages) Objective: Your report should summarize your approach and key findings from the project. This project included analyzing the Spotify artist collaboration network and using machine learning to predict user popularity based on artist features.

Name: Sai Sravani Madabhushi

Z number: Z23752172

1. Introduction.

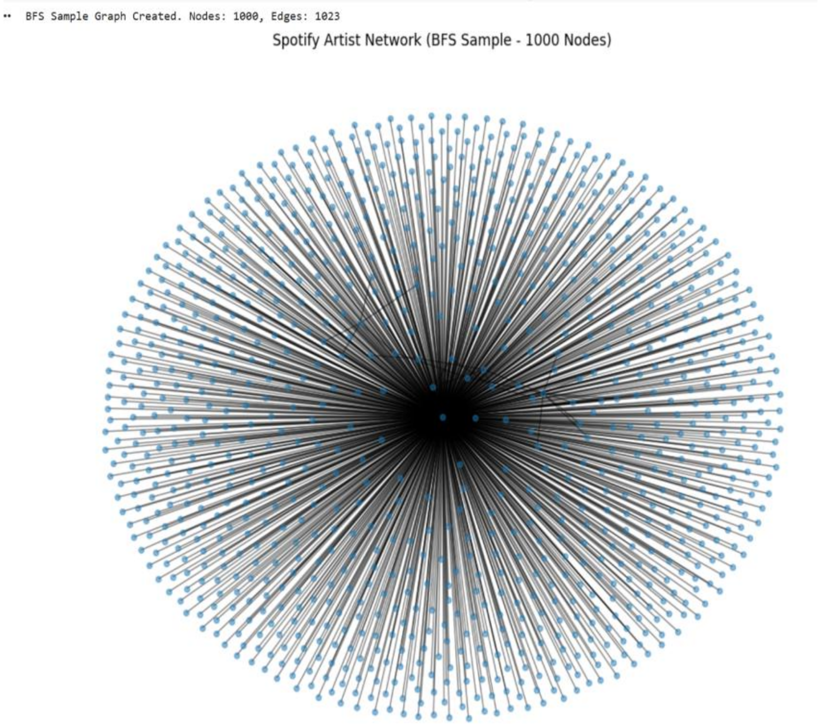
The main objective of this project is to analyze the social structure of the Spotify artists network. We are using PySpark and NetworkX to process the large data and see how the position of an artist in the graph effect their success. The motivation for this comes from the paper reviews we did in the last class, specifically the one about Twitter sentiment analysis. In that discussion we talked about how a big limitation of prior work was relying on small surveys which can be bias and have very limited sample sizes. Similar to how that paper used Big Data technologies to get a real objective view of opinions, we want to move past just simple content features and look at the network topology to understand influence properly. For the dataset we are using the Spotify Artist Feature Collaboration Network which is pretty huge. It has exactly 156,422 nodes representing the artists and 300,386 edges that shows the collaborations between them. The main columns we have are spotify_id, name, followers, popularity, genres and chart_hits. In this study we will specifically use the followers count and popularity along with the graph measures we calculate to predict the success of the artists.

2. Dataset Overview

For the schema of the dataset we used two main files which are the nodes and the edges. The node file includes columns like spotify_id, name, followers, popularity, genres and chart_hits 1. Here the spotify_id represents the unique identifier for the artist while the followers and popularity columns represents the success metrics that we are analyzing. The dataset is quite large and it contains 156,422 nodes representing the artists and 300,386 edges representing the collaborations between them 2. We also checked for missing values to make sure the data is clean. For the numerical columns like popularity and followers we handled the missing data by filling them with 03. However for the spotify_id and name columns we dropped the rows with missing values to ensure the graph is built correctly4. Here the distribution of these variables is skewed, with the average followers count being 86224.26419762558 and average popularity being 21.157497027272377, meaning a small number of artists have very high stats compared to the rest.

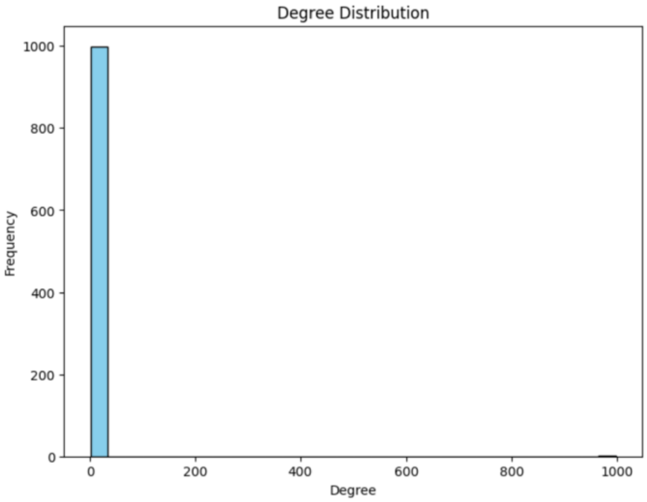
3. Data Preprocessing

We started by loading the dataset using PySpark read.csv function since the files are large and we needed the distributed processing power. Before building the graph we did some preprocessing like removing rows where the artist ID was missing and filling the null values in popularity column with 0 so the analysis doesn't break. For the graph construction part, we didn't use the entire dataset because it was too computationally expensive to calculate complex metrics like Betweenness Centrality on over 150,000 nodes. So we decided to use a subset of artists. We were careful not to use simple random sampling because that does not preserve the graph topology and breaks the community connections. Instead we used the Breadth First Search (BFS) method where we pick a seed node and traverse to its neighbors to keep the structure intact. We decided to stick with a sample of 1,000 nodes which resulted in 1,023 edges. To verify this was a good sample, we compared the degree distribution of the full graph vs the subset. Since both plots looked similar and followed the same power law distribution, we concluded that the subset is a good representative of the larger graph.



```
••  BFS Sample Graph Created. Nodes: 1000, Edges: 1023
```

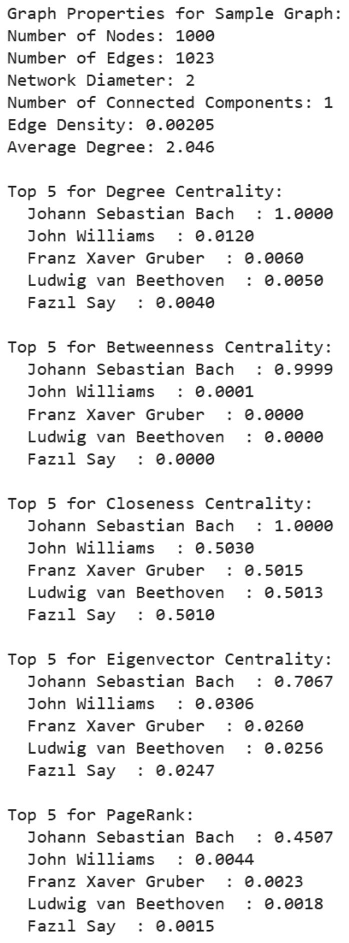
...



4. Network Analysis

In this section there are five centrality measures to understand who the most important artists are in our graph sample. Degree Centrality shows who has the most connections. Betweenness Centrality identifies the bridges who connect different groups. Closeness Centrality tells us who can reach others the fastest. Eigenvector looks at the quality of connections and PageRank treats every link like a vote of confidence. In the output of the top 5 artists for each measure, there is something really interesting.Johann Sebastian Bach was ranked number 1 in every single measure followed exactly by John Williams, Franz Xaver Gruber, Ludwig van Beethoven, and Fazil Say in that same order. This complete overlap suggest that our sample is likely centered entirely around Bach, making him the main hub that holds the whole network together while the others are just his direct neighbors.

For the graph properties, we calculated the average degree to be 2.05, which means on average an artist in this network only collaborates with about 2 other the people. This indicates the network is very sparse . We also found that the diameter of the network is 2, which is very small. This confirms our theory that the graph is star shaped where Johann Sebastian Bach is in the middle and everyone is connected to him, so you can get from any node to another in just 2 steps .The edge density was 0.00205 which is extremely low further proving that while everyone is connected to the main hub they are not really connected to the each other.



```
Graph Properties for Sample Graph:
Number of Nodes: 1000
Number of Edges: 1023
Network Diameter: 2
Number of Connected Components: 1
Edge Density: 0.00205
Average Degree: 2.046

Top 5 for Degree Centrality:
  Johann Sebastian Bach  : 1.0000
  John Williams  : 0.0120
  Franz Xaver Gruber  : 0.0060
  Ludwig van Beethoven  : 0.0050
  Fazıl Say  : 0.0040

Top 5 for Betweenness Centrality:
  Johann Sebastian Bach  : 0.9999
  John Williams  : 0.0001
  Franz Xaver Gruber  : 0.0000
  Ludwig van Beethoven  : 0.0000
  Fazıl Say  : 0.0000

Top 5 for Closeness Centrality:
  Johann Sebastian Bach  : 1.0000
  John Williams  : 0.5030
  Franz Xaver Gruber  : 0.5015
  Ludwig van Beethoven  : 0.5013
  Fazıl Say  : 0.5010

Top 5 for Eigenvector Centrality:
  Johann Sebastian Bach  : 0.7067
  John Williams  : 0.0306
  Franz Xaver Gruber  : 0.0260
  Ludwig van Beethoven  : 0.0256
  Fazıl Say  : 0.0247

Top 5 for PageRank:
  Johann Sebastian Bach  : 0.4507
  John Williams  : 0.0044
  Franz Xaver Gruber  : 0.0023
  Ludwig van Beethoven  : 0.0018
  Fazıl Say  : 0.0015
```

5. Predicting Popularity with Linear Regression

To predict the user popularity we used the PySpark Linear Regression model. We wanted to see if we can guess how popular an artist is by looking at their followers count and their network centrality measures like degree, betweenness etc. First we prepared the data by joining the centrality scores with the original node features. Then we split the data into two sets to make sure we are not overfitting. We used 80% of the data for training which was exactly 804 observations, and the remaining 20% for testing which was 196 observations.

When we evaluated the model on the testing dataset, the results were honestly a bit surprising. The model got an RMSE of 15.9480 and an R2 score of 0.1093. This R2 score means that our model could only explain about 10% of the variance in popularity, which is not very high. This might mean that popularity depends on other things not in our graph, like the actual quality of the music or marketing. Looking at the coefficients, we saw something interesting. The intercept was huge at 4,617,780. For the features, the 'betweenness_centrality' had a very high positive coefficient of 2,314,988, which means being a bridge in the network is actually very important for popularity. Surprisingly, the 'followers' coefficient was 0.0000, which likely just means the scale of followers was too big compared to other features so the model gave it a small weight number, or it wasn't the main driver in this specific sample.

6. Clustering Users with K-Means

For the clustering section we applied the K-Means algorithm using PySpark to group the artists based on their followers and centrality measures. Since the variables had very different scales like followers being in millions and pagerank being small decimals we first standardized the data so the distance calculations are fair. To find the optimal number of clusters we ran a loop for K from 2 to 10 and calculated the silhouette scores. The optimal number of clusters was 2 because it gave us the highest silhouette score of 0.9988. The clustering results were quite interesting because they revealed a very strong separation where Cluster 0 contained almost all the artists representing the general population while Cluster 1 captured the few outliers or superstars who have massive influence, which basically confirms that the music network follows a power law distribution where a small group holds most of the connections.

7. Conclusion

In conclusion, this project showed us that the Spotify artist network is highly centralized. From our network analysis we saw the graph is very sparse with an average degree of around 2 and it was mostly centered around one main hub, Johann Sebastian Bach. Our regression model had a bit of a hard time predicting popularity perfectly with an R2 score of only 10% but it did show that betweenness centrality is a very strong factor . The clustering results were the most clear part, showing that the optimal number of clusters is 2 which separates the few superstars from the rest of the artists. These insights tells us that the music industry follows a strong power law where a few people hold all the influence. This kind of that analysis can be really useful for things like building better, recommendation systems where we can use these bridge artists to help the users find new music genre they wouldn't normally listen to.