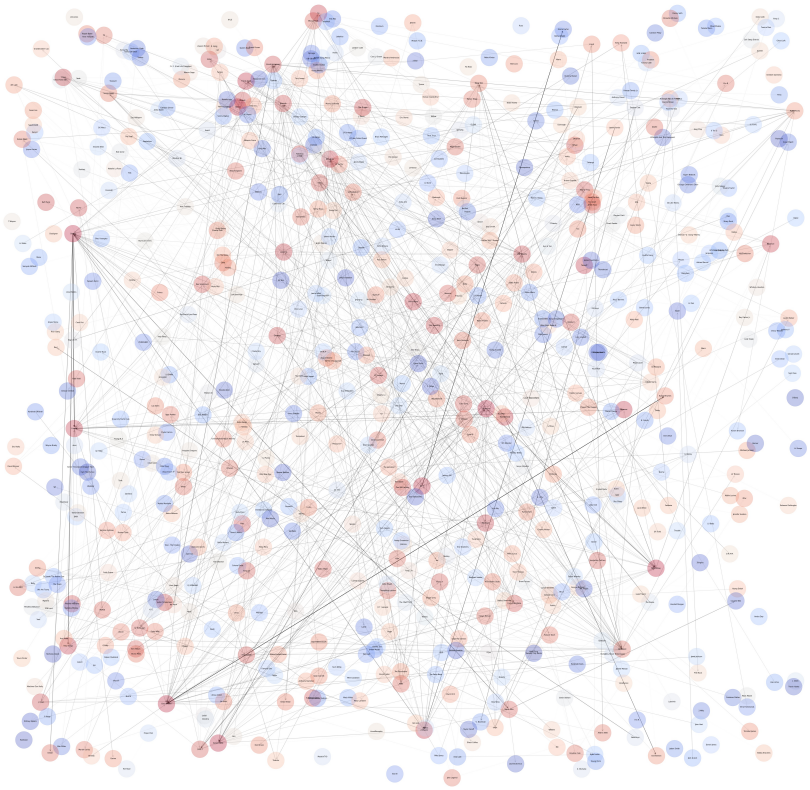
University of California, Berkeley
College of Engineering
Department of Electrical Engineering and Computer Science
Spring 2018

CloutRank

EECS126: Project Report – Markov Chains: PageRank and MCMC



Professor : Kannan Ramchandran
Students : James Fang
Rodrigo Henriquez-Auba
Matthew Jrke
Rishi Puri
Date : 04/18/2018

clout /klout/ NOUN
1. *informal* A heavy blow with the hand or a hard object.
'a clout round the ear'

2. *informal* Influence or power, especially in politics or business.
'I knew she carried a lot of clout'

Oxford Dictionaries

1. Introduction

The goal of this project is to rank R&B/Hip-Hop artists in terms of their collaborations with other musicians using the PageRank algorithm. We present several approaches to generating a PageRank graph using Billboard chart data and cross-validate these models using Spotify ranking data. Lastly, we employ our model to predict the success of new collaborations.

2. The PageRank Algorithm

In simple terms, the PageRank algorithm is used to obtain a ranking on how important a website is based on which other websites link to it. The algorithm outputs a probability distribution π on the set of webpages W, where $\pi(w)$ specifies the probability of a *random surfer* reaching a particular website. The intuition is that an important website should have a higher probability of a random surfer reaching it.

In this project we employ the PageRank algorithm to obtain a ranking on an artist A_1's importance (their *clout*) based on how often other artists $A_2, ..., A_n$ choose to collaborate with or feature A_1 and how much clout these other collaborating artists have. The corresponding intuition is that the average music listener M should have a higher probability of discovering artist A if A collaborates with other important artists that M listens to. The artists with the highest probability of exposure are the artists that hold the most clout in the music industry. In section 3. we describe how the relationship between artists is established.

For this project we will use the function `networkx.pagerank(`G`,`α`,pers_dict)` [1] that takes our graph G and computes its PageRank. The algorithm follows a random walk procedure which runs until convergence (or failure). This algorithm is different in some important regards from the simplified PageRank algorithm learned in class, which computes the stationary distribution of a Markov Chain constructed from a graph of links. Crucial to our application, the random surfer algorithm allows for weighted edges, self-loops and does not require the Markov chain to be irreducible.

The damping factor α represents the probability that a *random surfer* will eventually stop clicking links and then request another random page (for example directly typing in a new URL rather than following a link) [2]. In particular, this is used to prevent sink nodes absorbing the rank of pages connected to those sites. PageRank theory states that when a *random surfer* arrives to a sink node (in this case to an artist with no outgoing connections), the process terminates. With no damping factor ($\alpha = 1$), it is possible that the PageRank algorithm does not converge. We decided to use $\alpha = 0.85$ (recommended value in [1]), yielding a random jump probability of 15%.

At each step in the random walk, the chain evolves by either choosing a random outgoing edge from the current node or by jumping to any random node in the graph. By default it jumps to a random node according to a uniform distribution. The personalization dictionary `pers_dict` allows us to specify a different distribution. Artists that are more important, based off some chosen metric defined in section 4.1., should have higher weight in our distribution. The idea is that this personalized distribution represents our "prior" rankings of the artists or how we think the average music listener would rank the artists.

3. Model

Our data, obtained from the Billboard charts using a Python API [3], contains information on the top 50 songs every week for the past 10 years. For each of these songs, we're interested in the artists and featured artists on the song, as well as the peak position attained on the charts and the number of weeks the song was on the charts.

We construct a graph $G = (V, E)$ from the Billboard data, where each unique artist is a node in the graph. In addition, we specify an optional weight metric $w : E \to \mathbb{R}$, which is used to construct edge weights and our personalization vector (see section 4.1.). We outline several different approaches for generating the edge list

- We add an edge from artist A to artist B if B has been featured in a song by A. For example, consider the song *No Long Talk* by *Drake* featuring *Giggs*. In this case, we draw an edge from *Drake* to *Giggs*.

  This model reflects our assumption that artist A distributes a portion of their clout to artist B if A chooses to feature B on their song.

- (**Option 1**) If multiple artists appear on the same track and are not "featured", then we can draw double directed edges between every pair of artists on the track. For example, consider the song *Finesse* by *Bruno Mars* & *Cardi B*. Since this song has no "features", we draw an edge from *Bruno Mars* to *Cardi B*, and draw another edge from *Cardi B* to *Bruno Mars*.

  In addition to the assumptions outlined above, this model reflects that if artists A and B are collaborating on a song, both A and B should distribute some of their clout to their collaborators. Collaboration reflects equal standing, whereas a feature reflects hierarchy.

- (**Option 2**) If artist A has a song with no collaborators and no features, we add a self-loop $A \to A$. For artists that have many solo hits and few features, the self-loops ensure that their clout is retained.

Note that the two options are not mutually exclusive, i.e. we can include both options, just one, or none in our graph implementation. A example of how we construct a matrix is provided on the Appendix 8.1.

4. Methods

We analyzed the Billboard R&B/Hip-Hop charts from the last 10 years. Billboard lists the top 50 R&B/Hip-Hop songs, updated weekly. We chose to focus on R&B/Hip-Hop since features and collaborations are more common in this genre and the data yielded for it is the richest. The richness of the data is of concern, since sparse graphs may not converge in the PageRank algorithm.

Using a Python API for Billboard Charts [3], we retrieved the top 50 weekly songs of R&B/Hip-Hop genre from 04/19/2008 to 04/7/2018. We retrieved 1942 different songs from 655 different artists. Using a Python API for Spotify's library [4], we obtained a 'popularity score' from 0 to 100 for each artist, which Spotify calculates using listener counts. The Billboard data is used to generate our model, while the Spotify data is used to cross-validate our weight metric and ranking accuracy.

Using methods described in Section 2., we obtain a ranking for each version of our model, with each of our edge generation options switched on or off:

- We always have links from main artists to featured artist. This yields 1212 edges.
- **Option 1**: Bidirectional links between collaborating artists. This option adds an additional 514 (unidirectional) edges.
- **Option 2**: Self-loops for songs without any collaborators or features. This option yields an additional 994 edges.

4.1. Weight Metrics

In addition, we consider weighting edges to improve the accuracy of our predictions. We calculate weights from the Billboard data, based off of a song's peak position and number of weeks spent on the charts. We present different functions $w(s)$ for computing the weight of a feature or collaboration given an input song s. We wish to study the impact of these different weight functions on the CloutRank rankings and compare those rankings to our prior rankings:

- **Metric 1**: $w_1(s) = (51 - \text{Rank}(s))^2 \cdot \text{Weeks}(s)$
- **Metric 2**: $w_2(s) = \text{Weeks}(s)/\text{Rank}(s)$
- **Metric 3**: $w_3(s) = (51 - \text{Rank}(s))^2 \cdot \log(\text{Weeks}(s))$
- **Metric 4**: $w_4(s) = 1$ (constant metric)

Popular songs will place higher in the charts and spend more weeks on the charts, yielding a higher weight for that particular edge. If a song generates multiple edges, the same weight is assigned to each of the edges. If multiple songs generate the same edge, the total weight on this edge is the sum of individual weights.

4.2. Personalization

In addition to weighting our edges, the chosen metric is also used to compute our prior ranking for the artists, i.e. the personalization distribution, as described in section 2.. Let S_A be the set of songs that include artist A, then

$$\text{rank}(A) = \sum_{s \in S_A} \text{metric}(s)$$

The personalization distribution is given by the normalized rank vector, capturing our best estimate of prior distribution.

5. Results and Analysis

We compare each combination of our model's options (featuring/collaboration, metrics, weights and personalization) against a specific goal: a ranking based on popularity. Table 1 describes the different cases we run for the PageRank algorithm:

$$\underbrace{\left\{\begin{array}{c} w_1(s) \\ w_2(s) \\ w_3(s) \\ w_4(s) \\ \text{unweighted} \end{array}\right\}}_{\text{edge weighting}} \times \underbrace{\left\{\begin{array}{c} \text{artist} \leftrightarrow \text{artist} + \text{artist} \rightarrow \text{features} \\ \text{artist} \rightarrow \text{features} \end{array}\right\}}_{\text{bidirectional links}} \times \underbrace{\left\{\begin{array}{c} \text{personalization} \\ \text{no personalization} \end{array}\right\}}_{\text{PageRank customization}} \times \underbrace{\left\{\begin{array}{c} \text{self-loops} \\ \text{no self-loops} \end{array}\right\}}_{\text{solo artists}}$$

Table 1: Different cases to run.

We define the **mean edit distance** between lists X and Y, where Y is a permutation of X as

$$\text{MED}(X, Y) = \frac{1}{|X|} \sum_{x \in X} \Big|\text{index}(X, x) - \text{index}(Y, x))\Big|$$

The MED intuitively gives the average difference in predictive ranking and actual ranking per artist. We use the MED to measure the difference between the CloutRank of a given model both the with precomputed ranks (defined in section 4.2.) and Spotify's ranking. The MED allows us to not only cross-validate the accuracy of different metrics, but also assess the impact of our model's options on our prediction accuracy.

5.1. Comparison with precomputed ranks

In our first analysis, we compare the rankings produced by all possible models, running all the cases depicted on Table 1. Each model configuration produces a unique ranking, which we compare to our prior rankings defined in section 4.1.. We note that we keep the metric functions constant–we always use the same metric value for the personalization vector and for our goal ranking.

Please reference our accompanying iPython notebook for the full analysis statistics, which were too long to report in this document. The key results are summarized listed below. All MED values should be compared to a baseline off ≈ 217, the MED between 2 randomly shuffled lists averaged across 10,000 trials.

- Across metrics 1-3, the configuration: weighted edges *on*, artist-to-artist links *off*, personalization *on*, and self-loops *off* produces the lowest MED to the goal ranking. For metric 4, turning all options on yields the highest score, closely followed by the configuration listed previously in second place. We note that all options turned on produce good rankings for all metrics, confirming our assumptions that our model options generally improve our rank accuracy.
- Metric 3 provides the smallest MED $= 50.4$ between our prior and CloutRank. However, metric 1 has a minimum MED of 52.8 and metric 2 has a minimum MED of 56.5. For our purposes, metrics 1-3 all have similar accuracy, with metric 3 yielding a marginal improvement. The constant metric (metric 4) performs much worse, with a minimum MED of 107.5.
- Using metric 3 in the optimal configuration, the top ten artists are given by

$$\text{Rank}(w_3) = \begin{bmatrix} \text{Drake} \\ \text{Lil Wayne} \\ \text{Chris Brown} \\ \text{Nicki Minaj} \\ \text{Future} \\ \text{Rihanna} \\ \text{Trey Songz} \\ \text{Rick Ross} \\ \text{Beyonce} \end{bmatrix} \qquad \text{CloutRank} = \begin{bmatrix} \text{Drake} \\ \text{Lil Wayne} \\ \text{Nicki Minaj} \\ \text{Kanye West} \\ \text{Chris Brown} \\ \text{Quavo} \\ \text{Jay Z} \\ \text{Travis Scott} \\ \text{Rihanna} \end{bmatrix}$$

  We notice the similarity between these rankings, only interchanging extremely popular artists. CloutRank focuses on collaboration as opposed to rankings on charts, which is why certain top artists who collaborate a lot appear on the CloutRank (e.g. Quavo, Travis Scott) but not on the other.
- The option with the largest impact on MED was personalization, which is to be expected since the personalization vector is also our goal in this setting. To account for this bias, we compare our metrics with Spotify data in the next section.
- After personalization, edge weighting has the second highest impact on rank accuracy. Artist to artist links improves accuracy marginally, but to a greater extent than self-loops.

5.2. Spotify's Popularity Rank

Our second analysis compares our CloutRank against the popularity ranking using Spotipy [4].

- All metrics seem to perform equally well, with the constant metric surprisingly yielding the lowest MED of 139.3. This seems to indicate that Spotify's ranking is not dependent on Billboard chart information

(which is captured in metrics 1-3), but that the collaborations and features alone capture enough information to estimate their ranking. For comparison, the MED between our prior ranking (i.e. the personalization dictionary) and Spotify's ranking is 157.1.

- For all metrics, the optimal configuration appears to be: weighted edges *off*, artist-to-artist links *on*, personalization *on*, and self-loops *off*. In general, personalization and artist-to-artist links seem to improve performance the most.
- We note that our model has much higher MED's with Spotify's popularity rank compared to our own rank, although always performing better than baseline accuracy. Despite poorer accuracy across the board, this analysis allows us to validate our use of a proper personalization vector, which was our goal for this portion of the analysis. The use of a personalization vector tends to improve accuracy even against a different rank metric, so we are justified in using it.

In many ways, the Spotify popularity ranking is a poor parallel to CloutRank, so insights into model configurations (except for personalization) based on the MED to Spotify's ranking should be taken with a grain of salt. Spotify takes temporal factors into account, which cause artists who collaborated heavily 7-10 years ago to seem irrelevant. In contrast, CloutRank is temporally blind and would still give such an artist a high rank. Moreover, an artist's drive to collaborate and gain fame is not a well correlated predictor of monthly listens on Spotify. Spotify's ranking is also not limited to a specific genre. Certain pop artists only appear on a few hip-hop songs and received a low CloutRank, but because they are extremely popular in other genres they ranked quite well in the Spotify ranking, such as *The Weeknd* and *Ed Sheeran*.

6. Prediction

In the final stage of our project, we use our completed model to make novel predictions for artist collaboration. Suppose we have an aspiring artist A trying to break into the music industry. A only has a budget for one feature and wants to maximize their *clout* ranking based off this one feature. Which artist do they want to be featured by? Graphically, we have a new node representing the aspiring artist A. We can add one in-edge to A. Who should be on the other side of this edge?

For each of the 655 other artists $b \in B$ in our graph, we compute A's ranking if we only add the edge (a, b). We weight edge (a, b) with the average edge weight for artist b. We choose the artist such that A's ranking was the highest in the resulting CloutRank. Using this procedure, the top three artists to collaborate with are

1. **Takeoff**, yielding rank 22
2. **Pharrell**, yielding rank 33
3. **Quavo**, yielding rank 42

7. Conclusion

We have proposed a process for ranking rappers based on musical collaboration and analyzed how well this correlates to different metrics of popularity. This report shows a technical distinction between "clout" and the typical understanding of fame based on chart ranking and listeners. Our model ranks artists more on their ability to expand their network and collaborate with other artists rather than on the public's perception of them or their marketability.

References

[1] NetworkX, "Pagerank — NetworkX 1.10 Documentation," https://networkx.github.io/documentation/networkx-1.10/reference/generated/networkx.algorithms.link_analysis.pagerank_alg.pagerank.html, Oct. 2016.

[2] S. Brin and L. Page, "The anatomy of a large-scale hypertextual web search engine," *Computer networks and ISDN systems*, vol. 30, no. 1-7, pp. 107–117, 1998.

[3] A. Guo, "Billboard-charts: Python API for Billboard charts," https://github.com/guoguo12/billboard-charts, Apr. 2018.

[4] P. Lamere, "Spotipy - A Python Client for The Spotify Web API," https://github.com/plamere/spotipy, Apr. 2018.

8. Appendix

8.1. Example

Let us consider the following 7 songs with different artists (for simplicity, assume no features in this example):

Song	**Artist 1**	**Artist 2**
Congratulations	Post Malone	Quavo
Love	Kendrick Lamar	Zacari
Loyalty	Kendrick Lamar	Rihanna
I Get the Bag	Gucci Mane	Migos
Both	Gucci Mane	Drake
Kelly Price	Migos	Travis Scott
Big Shot	Kendrick Lamar	Travis Scott

Table 2: 7 songs example.

In these 7 songs, we have 9 different artists. Each song has multiple artists (i.e. mutual collaborations) but no features, and we draw double edges between artists according to option 1. Given these modelling assumptions we construct the PageRank transition matrix as follow:

Artist	Post Malone	Quavo	Kendrick Lamar	Zacari	Rihanna	Gucci Mane	Migos	Drake	Travis Scott
Post Malone	0	1	0	0	0	0	0	0	0
Quavo	1	0	0	0	0	0	0	0	0
Kendrick Lamar	0	0	0	1/3	1/3	0	0	0	1/3
Zacari	0	0	1	0	0	0	0	0	0
Rihanna	0	0	1	0	0	0	0	0	0
Gucci Mane	0	0	0	0	0	0	1/2	1/2	0
Migos	0	0	0	0	0	1/2	0	0	1/2
Drake	0	0	0	0	0	1	0	0	0
Travis Scott	0	0	1/2	0	0	0	1/2	0	0

Table 3: PageRank Matrix

Iterating over $(A^n)^\top x$, where x is a vector of ones, we obtain that Kendrick Lamar is the most important artist in this example:

$$(A^{100})^\top x = [1,\ 1,\ \mathbf{3/2},\ 2/3,\ 2/3,\ 4/3,\ 1,\ 1/2,\ 4/3]^\top$$