Automated Playlist Generation

Kade Keith, Demetrios Fassois
{kade, dimifass}@stanford.edu
CS 229, Fall 2017

MOTIVATION

With the growth of music streaming services, there are now more songs than ever at music listeners fingertips. Because of this growth, the art of constructing playlists has become increasingly challenging, and discovering new music the in the expanse of choices is a daunting task. We explore multiple approaches to, given some set of songs as a seed, deciding which other songs belong on the playlists based on lyrical and audio features. These models are trained on and evaluated against human-generated playlists.

DATA & FEATURES

Our data comes from the Million Song Dataset and musiXmatch lyrics collection. We augment those songs with info from Spotify. Our playlists are from Spotify.

Raw features:

- Tempo, Timbre, Danceability, Energy, Loudness, Speechiness, Acousticness, Instrumentalness, Liveness, Valence (Audio Features)
- Popularity, Year (Song Metadata)
- Lyrics in bag-of-words format

Derived features:

- TODO (Stuff from Timbre)
- TODO (LDA)
- TODO (Sentiment)

We chose our features based on what we found important while making our own playlists, and thus what we believe is important to others.

MODELS

Classification-based

Our primary approach was to think of selecting songs for a playlist as a classificaion problem.

Logistic Regression

Lorem ipsum dolor sit amet, mutat quodsi usu ei, ne quis nullam eruditi pro. Doming menandri ex sea, mel et sonet corpora phaedrum, per at veri porro neglegentur.

TODO

Lorem ipsum dolor sit amet, mutat quodsi usu ei, ne quis nullam eruditi pro. Doming menandri ex sea, mel et sonet corpora phaedrum, per at veri porro neglegentur.

Graph-based

An interesting alternative is to think of songs as nodes on a completely connected graph.

k-Nearest Neighbors

We represent each song as a point in n-dimensional space according to our normalized features. Then we select the next songs for that playlist based on proximity to the seed.

Shortest path (of length k)

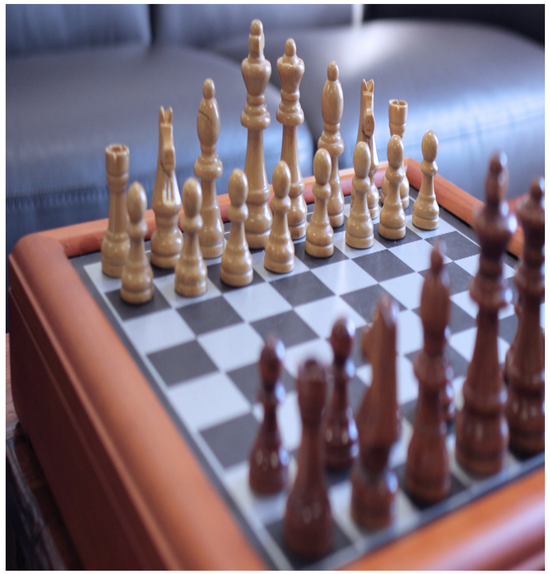
Given two songs, we construct a playlist as the shortest path of length k between them. This is particularly intriguing if the two seed songs are very different. Via dynamic programming, the shortest path to node v of length k is:

$$path[v][k] = \min_u \left(path[u][k-1] + dist[u][v] \right)$$

RESULTS

We trained and tested our models against X of popular playlists on spotify by splitting individual playlists into test/train sets. We used a random sampling of songs not on the playlist for negative examples. We made the decision around which playlists to use based on how many of the songs from the playlist we had data on.

For the classification approach, we report what percentage of the latter half of the playlist was correctly classified, and what percentage of a random selection of song was incorrectly put on the playlist. For the graph approach we report distance from our predicted songs to the songs we trained on, to the remaining songs on the playlist, and to a random selection of songs.



DISCUSSION

Lorem ipsum dolor sit amet, mutat quodsi usu ei, ne quis nullam eruditi pro. Doming menandri ex sea, mel et sonet corpora phaedrum, per at veri porro neglegentur. Imperdiet euripidis aliquando vis ea, ut nonumy quaerendum sed, at noluisse corrumpit vix. Sed reque tamquam deleniti ut, quo te simul decore convenire. Vix facilisis cotidieque ad, cu simul accusamus pro. Ne invenire contentiones vim.

FUTURE

Lorem ipsum dolor sit amet, mutat quodsi usu ei, ne quis nullam eruditi pro. Doming menandri ex sea, mel et sonet corpora phaedrum, per at veri porro neglegentur.

REFERENCES

1. Million song dataset. https://labrosa.ee.columbia.edu/millionsong/.
2. Spotify. https://www.spotify.com/.
3. TODO
4. TODO
5. Masoud Alghoniemy and Ahmed H. Tewfik. A network flow model for playlist generation. In In Proc IEEE Intl Conf Multimedia and Expo, 2001.