Predicting Popularity of Rap/Hip-Hop Songs

Temidayo Dairo
M.S. Mechanical Engineering
temidayo@stanford.edu

Hansub Kim
B.S. Mechanical Engineering
hkim808@stanford.edu

Nick Wilson
M.S. Mechanical Engineering
wilson19@stanford.edu

Abstract

Music is an influential and lucrative industry and being able to predict song popularity could prove to be highly impactful in such a competitive area. The phenomenon of popularity in music can be highly subjective so predicting popularity from audio features may potentially be more difficult than one perceives. Using audio data from Spotify, we attempted to predict the popularity of music within the rap/hip-hop genre.

1 Introduction

Music is a large part of just about any culture on the planet and is an emotional method of communication between human beings. Applications like Apple Music, Spotify, and Pandora grant listeners the access to millions of songs with a simple subscription and give people greater access to music than ever before. With the digitization of music and the creation of music streaming platforms, the social implications and lucrativeness of music have grown exponentially in recent history. Because of the large societal and financial impacts music has on world culture, our group decided to center our deep learning project on music.

For our project, our group wanted to develop a deep learning model that can predict the popularity of a song based on various audio features. The ability to predict song popularity would allow artists to test their music before releasing songs to listeners, giving the artist a better gauge of how their audience will respond to each song and ideally increase their popularity. Predicting song popularity using machine learning techniques has gained interest and effort from people within the community in recent years and has even been given a name - Hit Song Science. In this paper, we hope to differentiate ourselves from other ventures in this realm with our use of unique data acquisition techniques, deep learning algorithms, and datasets.

To give a better understanding of our project, we would first like to define the scope of our project. Because music is so diverse, we narrowed our project to only analyze modern songs within the rap/hip-hop genre. We believe that this genre will provide certain advantages for the application of deep learning with a repetitive and standardized beat. As Reiman and Örnell discovered in their research, audio features for songs have changed quite drastically over the last 70 years [5]. By narrowing our scope to only include data from modern rap/hip-hop songs we eliminate the variance in the audio feature data that may prove to be detrimental to the accuracy of our model.

With the scope of our project defined, we are able to identify the desired inputs and outputs of our model. We wanted the input to our model to be a list of audio features from a large set of songs and the output of our model to be an index for the popularity of the song. To acquire our dataset, we turned our attention to Spotify because it is one of the most popular music streaming applications and has an application programming interface (API) that can be used in python to gather detailed audio data. The Spotify API that we used allowed us to extract many audio features from a large list of songs within Spotify. The combination of all of these input features allowed our model to have a

CS230: Deep Learning, Winter 2018, Stanford University, CA. (LateX template borrowed from NIPS 2017.)

deep understanding of a song and was intended to allow it to make accurate predictions. The output of our model was the popularity of the song as determined by Spotify user statistics.

Once we had our dataset, it was time to develop a deep learning model. We tried a few different methods, but settled on a neural network of converging neuron count. We created our network to start with a relatively large hidden layer. This intuition was driven by the fact that we were limited by the relatively small size of our dataset. Each following hidden layer has a smaller number of neurons until the number of neurons converged to one single output. The only reason for gradually converging the number of neurons was to speed up the model. We believed that this would significantly reduce our computation time, thus allowing us to iterate more on our approach. We used and tested many hyperparameter configurations, as discussed below.

2 Related Work

Predicting song popularity is not a new practice, but one that we felt a great desire to try our hand at. We found prior examples of song prediction, using statistical models ranging from support vector machine models to recurrent neural networks. The most relevant models in Hit Song Science and their associated papers are discussed here.

In a paper by Michael I. Mandel and Daniel P.W. Ellis [3], popularity prediction is explored outside the realm of deep learning. We intentionally explored these approaches outside of deep learning to ensure that our problem was best suited for deep learning. With this in mind, we sought out to compare top results of prior statistical models outside of deep learning with that within deep learning. It turns out that models outside of the realm of deep learning can detect a signal better than a Gaussian algorithm. In work done by by Mohamed Nasreldin [4], six non-deep learning models were used for prediction including a support vector machine. These models ranged in accuracy from 0.513 to 0.632. Though these are certainly better than random, they are not useful for predicting the most popular song.

One group of Swedish computer scientists [5] attempted to predict the popularity of songs using machine learning. This group used many of the same audio features that we used as inputs to their model, however, they explore a variety of songs from many years, going back as far as 1950. Because the music data came from such a vast range of years, it was difficult for them to standardize their audio features. For example, the audio features showed a significant correlation with time over the course of many years. These struggles were a common theme that we saw in similar works. For this reason, we were inspired to select music from within a smaller time-frame, and within a single genre. These choices helped us maximize the standardization of our audio features.

3 Dataset and Features

We acquired our dataset using the Spotify API, Spotipy [2]. By using Spotipy, we were able to collect extensive data for each song that we used in our dataset. First, we manually created a list of 100 rap/hip-hop artists of varying popularity. Using Spotipy, we then got the Spotify track IDs for each song that each artist in our list is associated with. We were able to obtain 8,823 track IDs using this method. After we gathered all of the track IDs, we were able to use Spotipy to extract audio features for each song. A list of the audio features that we used as inputs can be found in **Table 1**. We also obtained the popularity index for each of the tracks to use as the output of our model. Spotipy calculates the popularity of a track with an algorithm that is based on the total number of plays the song has and how recent those plays are. Popularity index ranges from 0 to 100 for each song, with 100 being the most popular.

We also collected timbre and pitch data for each of the songs later in the project. This data was collected for multiple segments of each song and was intended to give a larger dataset for our deep learning model. The pitch of the song refers to how the notes of the song sound and is a perceptual property of sound that is related to frequency. Timbre is another perceptual property of music that measures the sound quality of a musical note or tone. Timbre is often referred to as tone color or tone quality and helps to differentiate between different sound sources.

Spotipy Feature Name	Type	Description
Key	int	Estimated overall key of the track
Mode	int	Modaility (major/minor) of a track
Time Signature	int	Estimated overall time signature of a track
Acousticness	float	Confidence measure of whether the track is acoustic
Danceability	float	How suitable a track is for dancing based on musical elements such as tempo, rhythm stability, beat strength, and overall regularity
Energy	float	Represents a perceptual measure of intensity and activity
Instrumentalness	float	Predicts whether a track contains no vocals
Liveness	float	Detects presence of an audience in the recording to determine if the track was recorded live
Loudness	float	Overall loudness of a track in decibels [dB]
Speechiness	float	Detects the presence of spoken words in a track
Valence	float	Measure from 0.0 to 1.0 quantifying the positiveness (e.g. cheerful, happy, euphoric) of a track
Tempo	float	Overall estimated tempo of a track in beats per minute [BPM]

Table 1: Descriptions for each of the Spotipy audio features that are used as inputs for our model

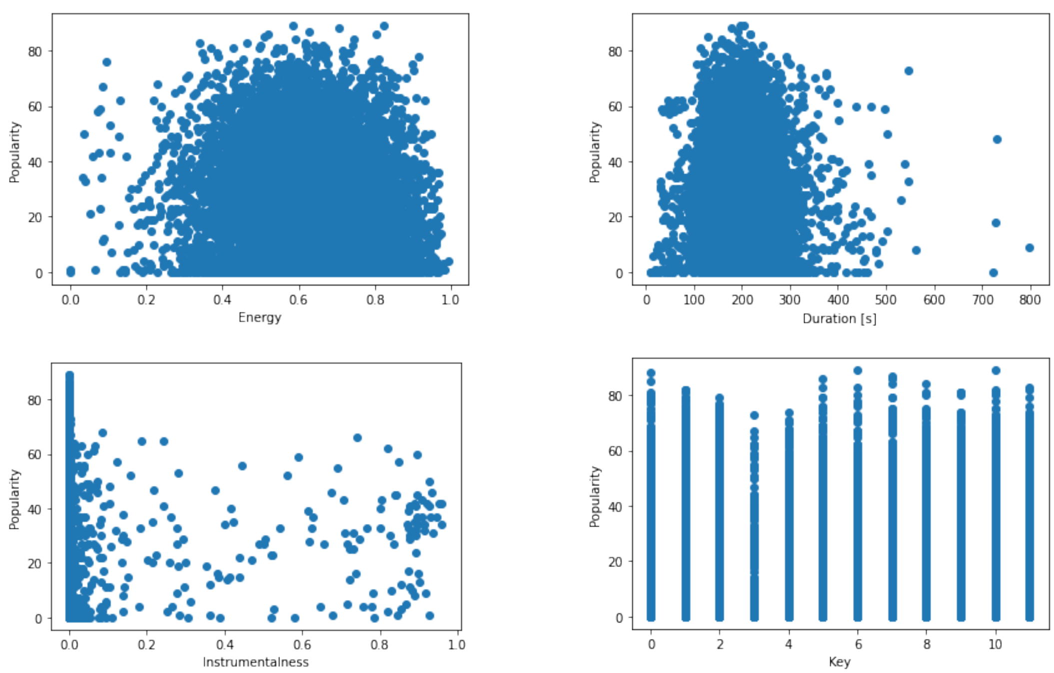

Figure 1: Plots of song popularity against various audio features for the 8,823 songs within our dataset

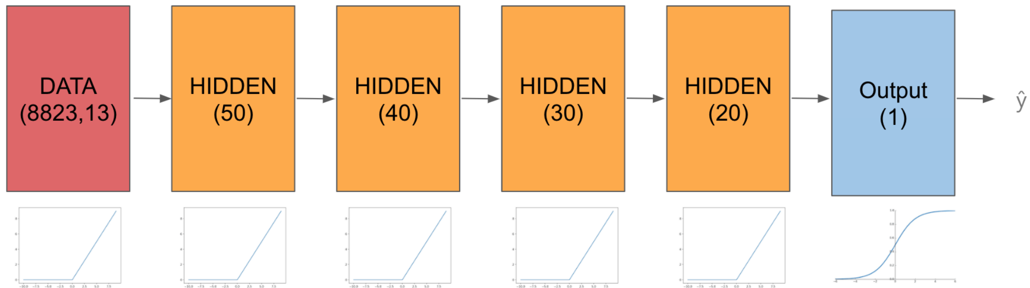

Figure 2: Neural network structure for our deep learning model. The input of our model consists of audio features from 8,823 songs. The hidden layers of the network decrease in size as we get deeper in the neural network. The output of the model is a single value for popularity.

4 Methods

We decided to use Keras [1] to build a deep neural network for predicting popularity. With the model parameters that we have chosen, including the artist's name, we are able to detect a signal that can correlate with track popularity. We are in the process of tuning the model in hopes of obtaining good results. We decided to consider the network depth and layer size as hyper parameters, and tried a few configurations of these parameters. With this in mind, we decided to start with 16 layers, each with 16 neurons. At first, the model required more depth, and eventually, the model performed best with 4 hidden layers, the first with 50 neurons, and each subsequent layer with then fewer neurons. Next, we expanded the width of each layer to see if the model improved. Another important hyperparameter is learning rate, which was tuned before the model depth and width. A high-level schematic of our deep learning model can be found in **Figure 2**.

5 Experiments/Results/Discussion

Our process of modifying and improving the model took an iterative approach. Using many different optimizers, loss functions, and activation functions, we were able to come to a conclusion as to which of each were best for our model. This was not entirely a final decision, but we felt that the model configuration that we chose was well suited for our application. Initially, our model gave values of error that held constant after numerous epochs. We saw this as a bad sign and made numerous changes to fix this, however these changes resulted in stationary oscillations about a similar mean. This was a hint that perhaps our dataset was too small. However, even after we increased our dataset size to include the timbre and pitch data for each song, the same problems that we experienced with the smaller dataset persisted. In fact, we continued to have the same results

We made use of many different optimization algorithms and loss functions. We found that by using mean squared error, we were able to improve our accuracy in comparison to binary cross entropy. We believe that binary cross entropy would be better suited for a classification application. Though we initially assumed that Adam would be used as the optimizer, we were proven wrong when we attempted to use stochastic gradient descent. Stochastic gradient descent worked better for our model, we hypothesize, because Adam was prone to falling into local minima. We used relu as an activation function for all layers except for the output layer. This is because we needed a nonlinear function that was defined for all input values. We considered tanh for these activations, but chose relu experimentally, after it performed better. Our output was well suited for a sigmoid activation because of its bounded nature. Because all values stayed between 0 and 100, we simply scaled them to between 0 and 1 so that they would fit the sigmoid output well.

Although we set out to produce a working deep learning model that could predict song popularity for rap/hip-hop songs, we were unable to attain results that were up to our standard. From **Figure 3** we can see that our model produced undesirable results. From the figure, we can see that the loss of our model remained at a constant value or oscillated about a singular value despite our many attempts to alleviate this problem. Although we were unable to produce results that did not reflect our expectations, we learned a great deal from the process. As we continually tried to debug our deep

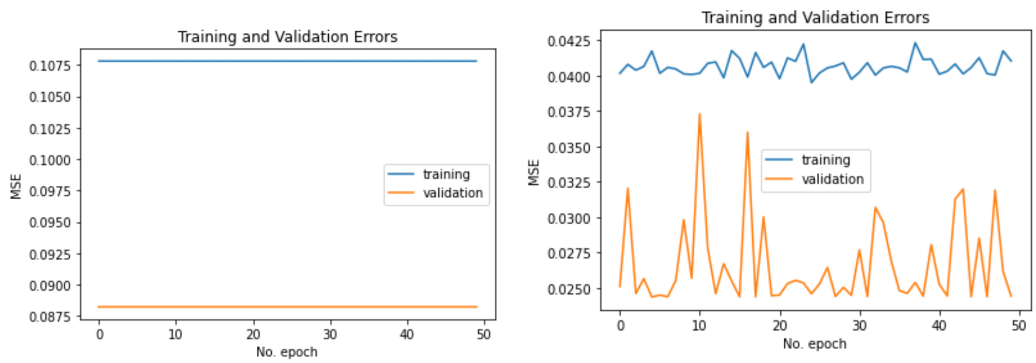

Figure 3: Plots of song popularity against various audio features for the 8,823 songs within our dataset

learning model, we found ourselves using techniques that were taught to us in the class throughout the quarter. We tried adjusting hyperparameters, loss functions, activation functions, and normalization techniques.

In this project, we really tried to apply our knowledge of deep learning from the quarter to solve the many problems that we encountered along the way. We were unable to obtain results like those that we saw from the literature that we reviewed, but we still believe we learned a lot and utilized the deep learning techniques we learned throughout the quarter to gain a deeper understanding and appreciation of deep learning neural networks. If

6 Conclusion/Future Work

We believe that a much larger dataset is necessary to obtain the results we had hoped for. Due to the lethargic nature of the Spotify API's request protocol, we were not able to build a sufficiently large dataset quickly enough. Prior literature struggled with the effects of chronology, as music is ever evolving. One day, when the throughput of music is much larger for a given year, the landscape will be more vast, and therefore more suitable for a model to predict popularity of rap songs. Just as in the prior literature, we find that although decent results are possible, the results are eventually limited by the rate of music production which limits the size of our contemporary dataset. Music is a highly subjective area of study in machine learning and we believe that it is difficult, but not impossible, to collect data that is sufficient to build a deep learning model on. In future work, we would like to explore larger datasets and even more options for our model. Music remains a highly influential form of media and as its implications grow, so will the attempts to harness its financial and cultural impacts.

7 Contributions

Temidayo Dairo: I worked on decomposing the problem and helped with creating functions for building the dataset. I also helped to write our deep learning code, and helped with our report and video.

Hansub Kim: I worked on helper functions for making requests to the Spotify API, and pre-processing for the dataset. I also helped build the deep learning model and set the Keras neural network up.

Nick Wilson: For this project, I worked on data acquisition and the reports and video. I spent most of my time creating functions for building the dataset, writing the final report, and producing the final video. I also helped build the deep learning model.

References

[1] Keras API Reference. https://keras.io/api/

[2] Lamere, Paul. “Welcome to Spotipy!” Spotipy 2.0 Documentation, 2014, spotipy.readthedocs.io/en/2.16.1/.

[3] Michael I. Mandel, Daniel P.W. Ellis. "Song-level Features and Support Vector Machines for Music Classification." New York: Queen Mary, University of London, 2005. https://www.ee.columbia.edu/ dpwe/pubs/ismir05-svm.pdf

[4] Nasreldin, Mohamed. “Song Popularity Predictor.” Medium, Towards Data Science, 14 May 2018, towardsdatascience.com/song-popularity-predictor-1ef69735e380.

[5] Reiman, Minna, and Philippa Örnell. “Predicting Hit Songs With Machine Learning.” 30 May 2018, http://kth.diva-portal.org/smash/get/diva2:1214146/FULLTEXT01.pdf.