Predicting Song Popularity on Spotify

Capstone Project

Preet Gandhi, Ksenia Saenko, Stella Sun, Wenjie Sun, Ben Zhang

New York University - Center for Data Science

Fall 2018

Introduction

Problem statement

Given a track before it is released to the public, how do we know if it would be a "hit" song on Spotify, which is the biggest music streaming platform that has 170 million worldwide users? Without any marketing strategy or promotion information, only from the track attributes: title, artist, producers, lyrics, melodies, is it sufficient to predict its success?

We are using machine learning techniques to use 14k Warner Music songs and lyrics and predict their popularity on Spotify. Our goal is to discover explainable patterns and actionable insights for Warner Music based on this analysis. The company's profits are dependent on the ability to find songs with a high potential to become popular in order to promote and support them. Yet identifying such songs is a challenging task.

Challenges

Early industry research showed that Music Science has been studied by many people in the past decades, however, no one has achieved good results on modeling music and predicting success.

In addition, we had limited access to data sources, such as the original mp3 music files, lyrics, artist and producer information, etc. We spent half of the time to get the data (either scraping the data or looking for an existing database) and also cleaning the data.

Therefore, our goal was to get as much data as we could in order to beat the industry average (AUC score never go above 60%), in which case we would say we have achieved a good result.

Data Sources

Audio Files

We received 15k mp3 files from Warner Music and we used these mp3 audio files to train a deep learning model.

Lyrics

Lyrics were used in topic modeling, sentimental analysis and repetitiveness analysis; we also built a CNN for processing lyrics as a separate model. All lyrics were scraped using Genius API.

Artists, Producers and Songwriters

Genius API provided some information about the producers and songwriters, however, we were only able to find 50% of the songs with producer and songwriter name(s) .

We separated artists into three levels, A, B and C, based on our interviews with the Warner team. A artists such as Drake, Taylor Swift, Ed Sheeran will always have a hit song once they release new music, so to avoid those songs, we excludes these artists. B level artists are referring to those who sometimes have hit songs, but sometimes not; C artists consisted of new artists or artists that never had a hit song. However, there could be some potential rising stars in C levels.

We featurized producers/songwriters as follows:
missing_producers: *binary*, if producer information is available
producer_count: *numeric,* how many producers
a_producers: *binary*, if any of the producers is an a_producer (we used a predefined a_producer list that consists of all producers who wrote hit songs for A-level artists)
a_producer_count: *numeric*, how many a_producers

We repeated the same procedure for songwriters.

Genres

The genre often determines one's preference for music. If we look at the Spotify's Viral Top 50 chart, we will see that the majority of it is comprised of R&B and Hip-Hop artists. On the other hand, we would never expect to see a classical or a jazz song appear on one of such charts. Therefore, it was important to control for the genre in our analysis.

To obtain genre data we used Spotify API which provides genre by artist ID. Interestingly, this way we obtained over 2000 different genres in our dataset. Spotify is very specific in terms of how it identifies genres. For example, we found some unique genres, such as "deep German hip-hop". Clearly, using all 2000 genres would create too much noise in the data. Instead, Warner Music provided us with a mapping of Spotify genres to

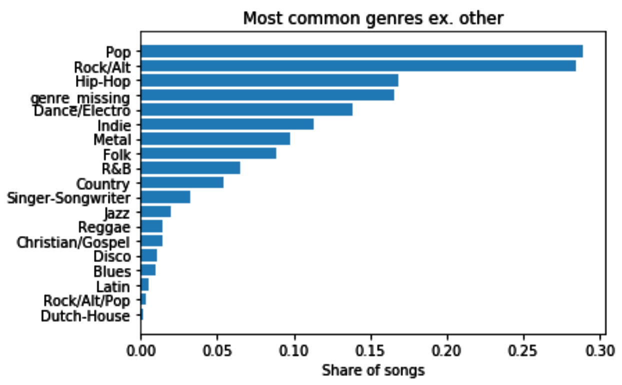


Figure 1: Song distribution by genre

the genres used by Warner internally. We performed the mapping and ended up with 20 most common genres. (Figure 1).

Labels

We looked at popularity score defined by Spotify, which normalizes for such attributes as a song release date. We picked the songs with a maximum popularity score above 70 as hit songs, and the labels were assigned as 1 if a song is a hit, otherwise, 0. The threshold of 70 was based on our discussions with Warner Music as well as data distribution analysis. This was a high bar, given that the vast majority of songs never achieve a popularity score above 40.

Methods

As we mentioned at the beginning, this project is not just about applying machine learning to predict song's success on Spotify, but also requires a lot of effort on getting data, data pre-processing, validation, and transformation. Feature extraction and engineering were crucial for achieving success in this project. For example, just using the Spotify data which was the dataset we started with in the beginning of the semester led to a 51% AUC at most, meaning the original dataset had no signal about song popularity.

In this section, we are going to share every step we took to get a finalized dataset, and the methods we used to model the data.

Data Processing and Feature Engineering

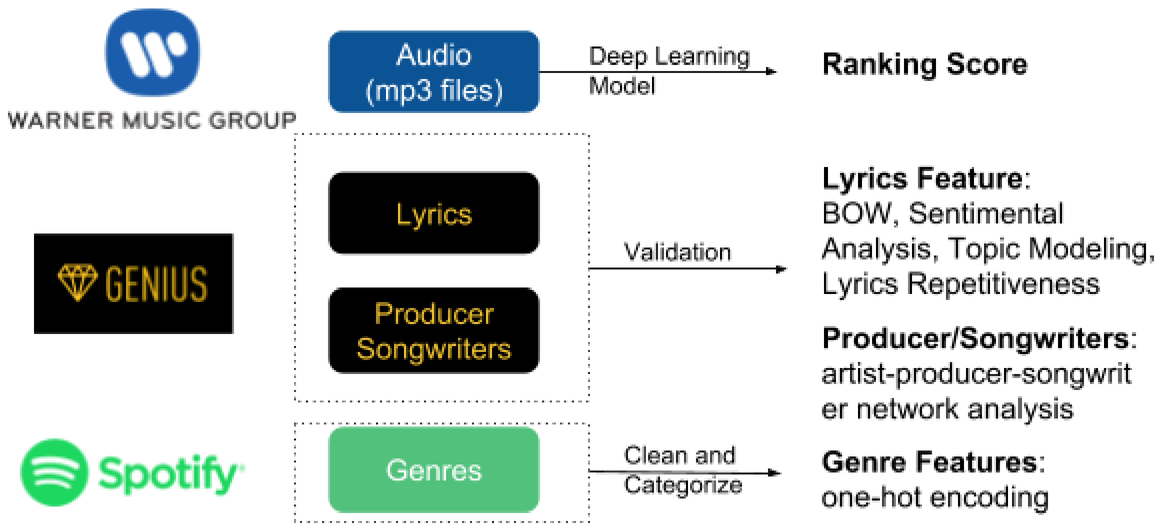

WARNER MUSIC GROUP
GENIUS
Spotify
Audio (mp3 files)
Lyrics
Producer Songwriters
Genres
Deep Learning Model
Validation
Clean and Categorize
Ranking Score
Lyrics Feature: BOW, Sentimental Analysis, Topic Modeling, Lyrics Repetitiveness
Producer/Songwriters: artist-producer-songwriter network analysis
Genre Features: one-hot encoding

Figure 2: Data Processing and Feature Engineering

As depicted in Figure 2, we obtained mp3 files with audio data from Warner Music. We scraped lyrics and producer, songwriter data using Genius API. We also obtained genre data using Spotify API, among other features.

Machine Learning Method

Since it is a classification problem, and we have three main data sources: audio features, lyrics features, all other supplement features (artists, producer team, etc.), we would try two neural networks on audio features and lyrics, and a classification model on other features. Deep learning has achieved great success in modeling language and audio data, unlike the traditional machine learning methods. For other one-hot encoded features we wanted to build a simpler model that could be easily explained for business purposes. For example, we wanted to be able to visualize the model decision process and explain the most important features, so that Warner Music could use this findings in a meaningful way. Therefore, we experienced with such models as decision tree and logistic regression, among others.

As a final model, we could combine the three models built separately using either linear model or some other method to improve the overall performance.

Results

We wanted to present both analytical results and machine learning model results because a clear analytical analysis would help Warner Music better understand their artists, songs and lyrics.

Analytical Results

Exploratory Data Analysis - Notable Findings

(1) We explored the data by looking at the distribution of the popularity score of each song in our final database. Most songs in our database are around 20 popularity score as expected. We decide to keep Warner's suggestion of choosing >70 to define our binary target, which is 6.5% popular songs of the total database.

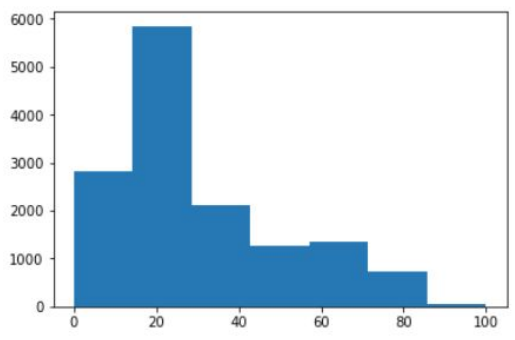


Figure 3: Distribution of popularity score

(2) We also explored clustering artists by their number of albums, number of tracks, number of playlists, and number of followers. By looking at the clustering, more albums or tracks doesn't correlate to a higher average popularity score.

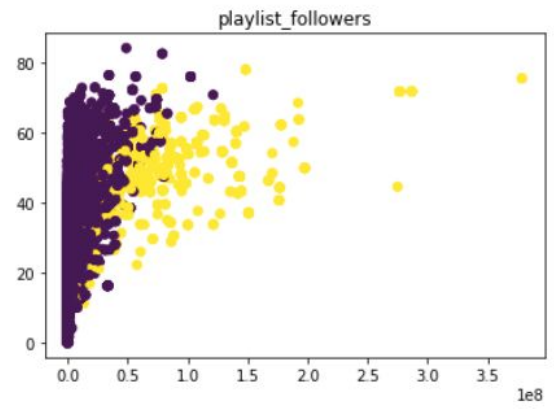
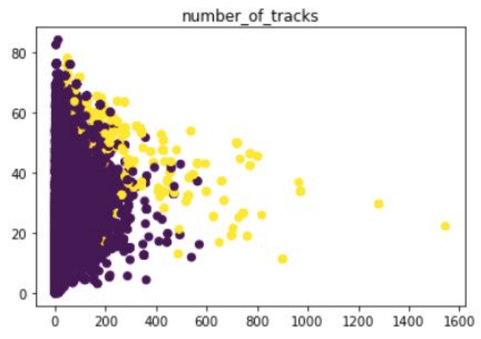


Figure 4 : Playlist followers vs popularity score

However, more playlist followers is correlated to artist's average popularity score. However, we believe there is a causation relationship and thus we decide to exclude this data it from future analysis.



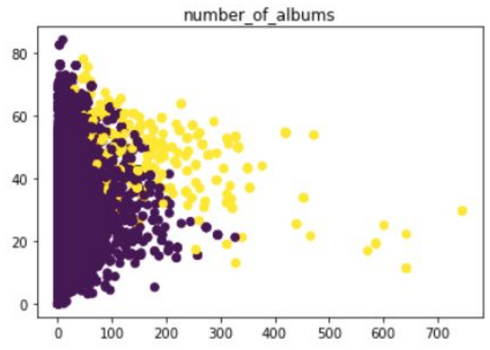


Figure 5: Number of tracks vs popularity score   Figure 6: Number of albums vs popularity score

Audio Feature Analysis

We extracted two features from audio data: Mel-Spectrum and Song Tags. Mel-Spectrogram is an acoustic time-frequency representation of a sound: the power spectral density $P(f, t)$. It is sampled into a number of points around equally spaced times t_i and frequencies f_j (on a Mel frequency scale). We extract 128-bin log-scaled mel-spectrograms with sampling rate 22,050 Hz and a half-overlapping hamming window of size 4,096 samples for short-time Fourier transform from the first 90s segment per song, using the librosa library. The first 90s of a song is enough for an audience to decide if he/she like this song or not.

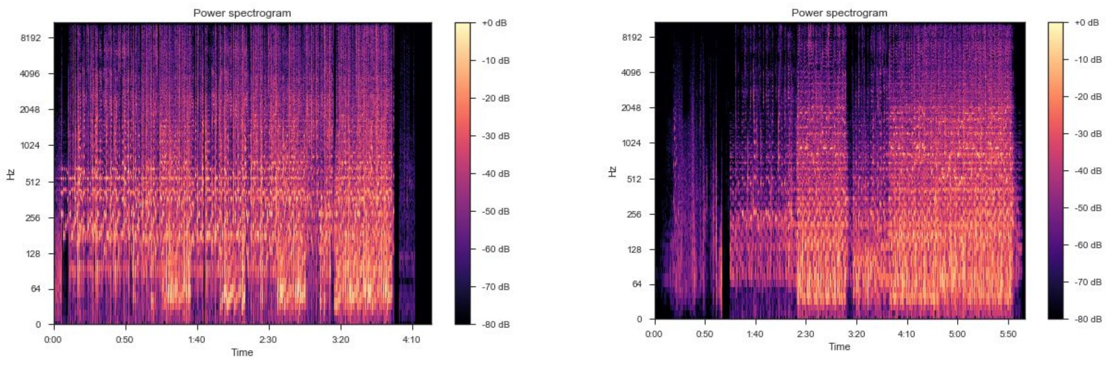


Figure 7: The mel-spetrogram of two examples (Left: Ed Sheeran - Shape of You; Right: Adele - Hello)

However, the mel-spetrogram is low-level features and may suffer from the “semantic gap” . So we use a pre-trained multi-label music tagging CNN model called JY-NET which can extract semantically rich tags (labels) such as instruments-guitar, piano, vocal, etc.Then we feed the songs of our dataset to this tagging model and take the top 50 tags as an additional input feature representation in our Siamese.

Sentimental Analysis on Lyrics

We also studied the sentimental analysis of lyrics. Since we don’t have pre-labeled sentiment in our data, we use a pre-trained R package (SentimentR). Overall, all the lyrics in our final database is neutral.

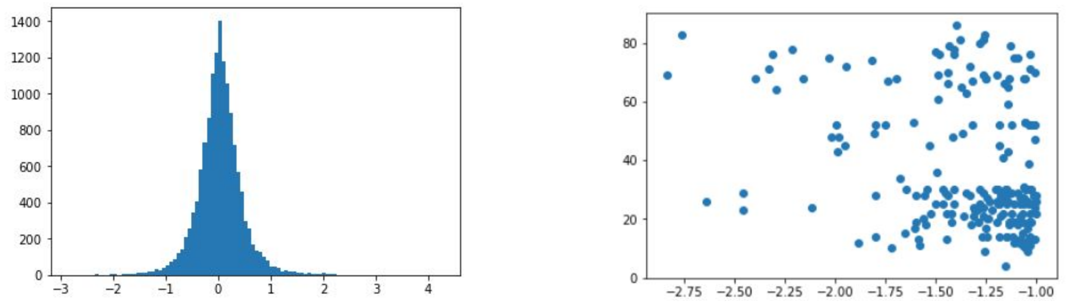


Figure 8: Sentimental Analysis

However, when a song is extremely negative (sentiment score is lower than -1), it is significantly correlated to the popularity score (p-value is 3.11e-5). The more negative the sentiment is, the more popular the song is.

Lyrics repetitiveness

To determine how repetitive a song is, we used Abraham Lempel and Jacob Ziv (LZ77) algorithm to identify repetitions in lyrics. The algorithm finds and compresses parts of a text that are the same. To obtain this information, we first compressed each file using GZIP - which uses DEFLATE, a combination of LZ77 and Huffman coding. Then we used the program called *infgen* which provided statistics of a file compression by gzip. We used the statistics on compression to estimate how repetitive a song is, based on how much it was compressed.

The charts below visualize lyrics repetitiveness using similarity matrix where cell (i, j) is filled if a word *i* in a song's lyrics is the same as word *j*. Each distinct word that appears more than once is depicted in a different color. So we can see that, for example, Cardi B's song " I Like It" has a lot of chorus repetition throughout the lyrics (Figure 9).

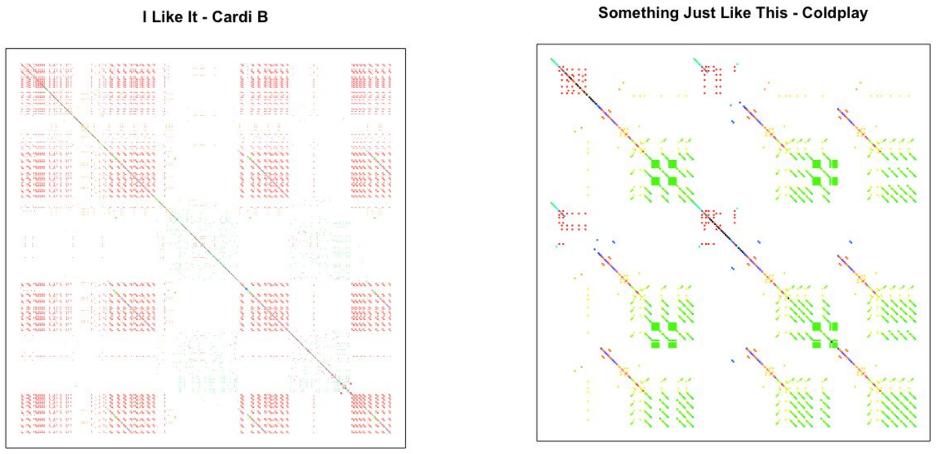


Figure 9: Lyrics Repetitiveness

Topic modeling

For each genre, we did topic modeling on the lyrics to find out which lyrics are similar or can be grouped together and what are the popular topics for those groups. We used the LDA implementation from Gensim package along with the Mallet's implementation (via Gensim).

Mallet has an efficient implementation of the LDA which runs faster and gives better topics segregation. We used pyLDAvis for interactive visualization. LDA's approach to topic modeling is it considers each document as a collection of topics in a certain proportion. And each topic as a collection of keywords, again, in a certain proportion. Once you provide the algorithm with the number of topics, all it does it to rearrange the topics distribution within the documents and keywords distribution within the topics to obtain a good composition of topic-keywords distribution.

We obtain different topics where each topic is a combination of keywords and each keyword contributes a certain weightage to the topic. A topic is nothing but a collection of dominant keywords that are typical representatives. Just by looking at the keywords, you can identify what the topic is all about. To get better quality of topics which can span multiple words, bi-grams and tri-grams were also used to improve the model.

In the below snapshot of the interactive representation, we see that in hip-hop genre, the topic 13 has words that seem to talk about drug-crimes and jail-time in the youth. (Figure 10)

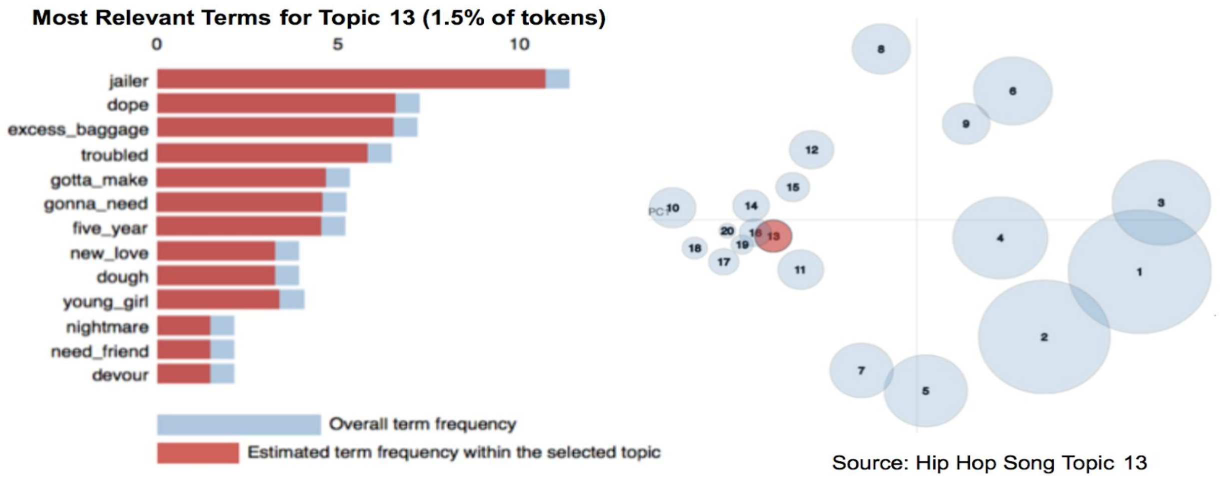
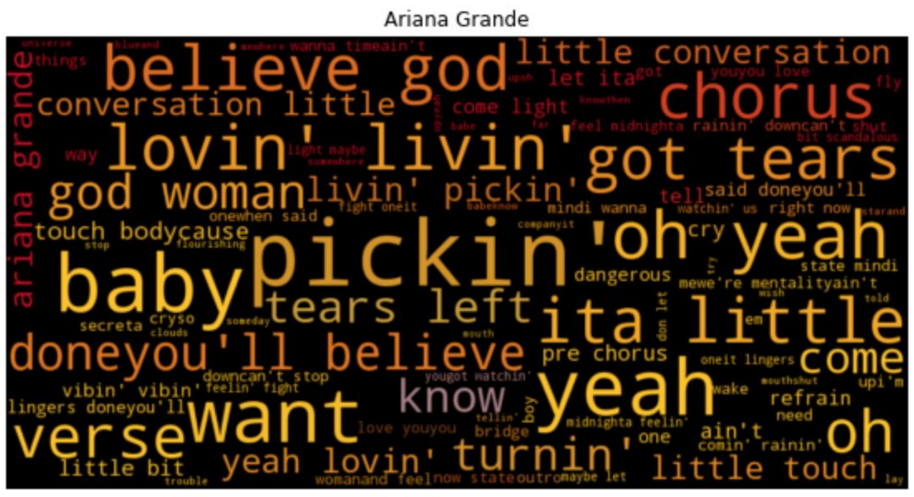


Figure 10: Topic Modeling

Interesting Findings on Lyrics

We also ran the topic modeling by artist, and here is the word cloud visualizations for Ariana Grande and Kendrick Lamar, where we can see a huge difference between their lyrics.



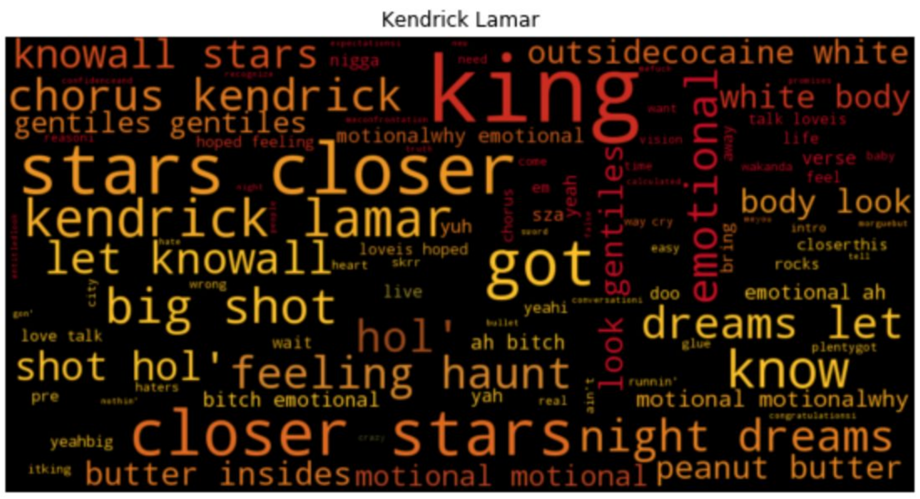
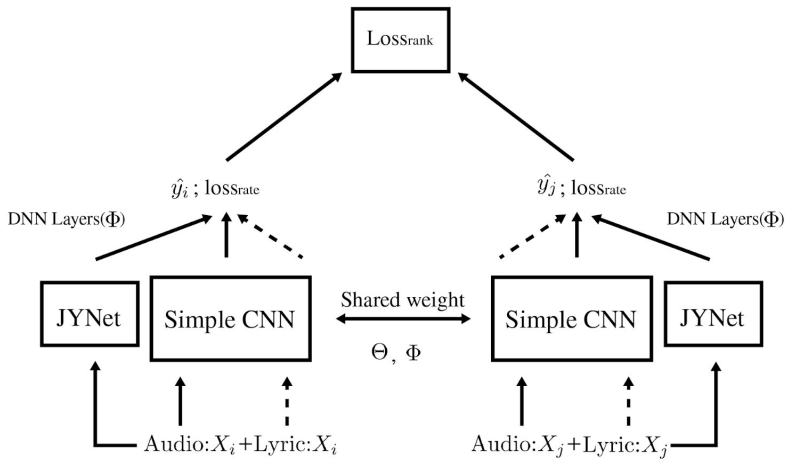


Figure 11: Topic Modeling Examples

Model Results

There are three models we developed: Siamese-CNN using audio features, CNN using lyrics, and Decision Tree using all other features. A final model took the predictions from all three models and combined the signals together to produce the final prediction.

Siamese-CNN on Audios



Siamese CNN with audio, lyric &tag features

Figure 12: The Siamese CNN model structure

As shown in Fig.12 , we built our Siamese-CNN which is composed of two identical Inception CNN model. They share the same parameter set Θ, Φ. Given input pairs (x_i , x_j) from the training set, we optimize Θ , Φ by minimizing the following pairwise ranking loss proposed in:

$$\text{loss}_{\text{rank}} = \frac{1}{P}\sum_{i,j} \max(0, m - \delta(y_i, y_j)(f_\Theta(\mathbf{x}_i) - f_\Theta(\mathbf{x}_j)))$$

Eq. 1 Pairs Ranking Loss Function

, where P denotes the number of song pairs we use, m > 0 is called the "margin" and is a hyper-parameter to be tuned by using the validation set, and δ(yi , yj) returns 1 if $y_i \geq y_j$ and −1 otherwise. Therefore, if $y_i \geq y_j$, Eq. 1 prefers that $f_\Theta(x_i) - f_\Theta(x_j) \geq m$. It encourages the model to rank the songs correctly (with a certain confidence that is related to m), and will penalize it if the model ranks wrong, without considering the actual difference between yi and yj. We can combine the two loss functions and have a multiobjective Siamese CNN $f_\Theta(\cdot)$ that minimizes

$$\text{loss}_{\text{multi}} = (1 - w)\ \text{loss}_{\text{rate}} + w\ \text{loss}_{\text{rank}}$$

Eq. 2 Ranking Loss Function

where w ∈ [0, 1] is another hyper-parameter to be decided from the validation data.

The main purpose of this model is to rank the hit songs by popularity, where a model can detect the hit songs among all other songs after shuffling the songs multiple times.

For present, we just trained the audio and tags, but in the future, we will incorporate the hyper extracted features from lyrics.

After running 80 epochs, training loss dropped from 374987 to 148573, and the model achieved 25% recall of the top 100 songs.

CNN on Lyrics

A pre-trained word embedding from FastText was used to embed the tokenized lyrics word; and a single layer Convolutional Neural Network was used to train the model on lyrics and predict the binary label.

After running 20 epochs, training loss converged, and the model achieved 33% recall.

Decision Tree on Other Features

We also used Spotify audio features (such as liveness, valence, dance, energy), song genre, information about the team working on the song (number of writers, number of producers), and engineered features from the lyrics (the sentiment of a song, and lyrics repetition) to build a tree-based model. The advantage of the model is its easy explainability and ability to visualize and extract business insights. After tuning hyperparameters, we achieved the best AUC of 67% with the following parameters: maximum depth of 20 and minimum samples split of 5. The most important features uncovered by the model are related to the team working with the artist, such as the number of writers and producers. Explicit lyrics, repetitiveness in lyrics, and Pop genre are also highly predictive.

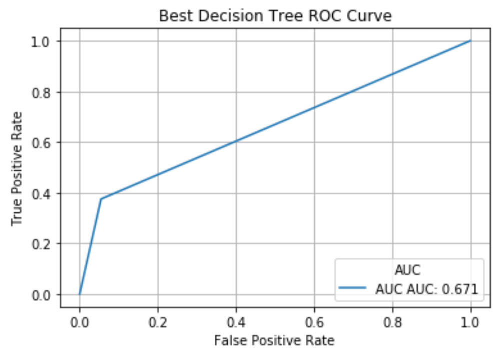


Figure 13: ROC Curve for Decision Tree

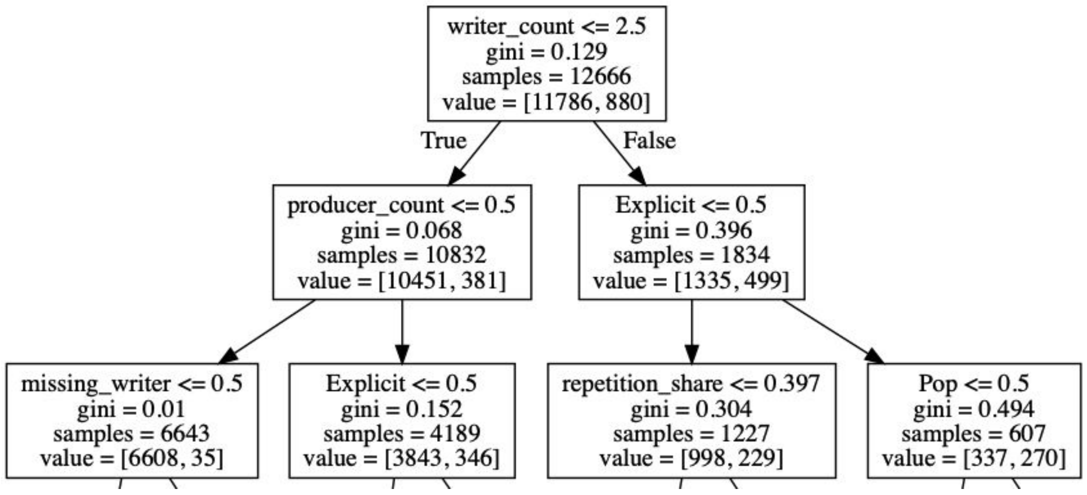


Figure 14: Decision Tree Result (max depth = 3 for visualization)

Final Model

We had to come up with a way to combine predictions from three separate models built on lyrics, audio data, and other song information. In order to do so, we experimented with three versions of stacked models. We tried predicting the majority class from three model outputs, logistic regression built on the three model outputs, and optimistic stacking, which led to the highest recall of 40%. Optimistic stacking predicts a positive class if any of the three predictions from the input models predict a positive class (Figure 15). This approach is logically appropriate because our goal is to achieve the highest recall possible - missing a potential hit is more costly than promoting false positives.

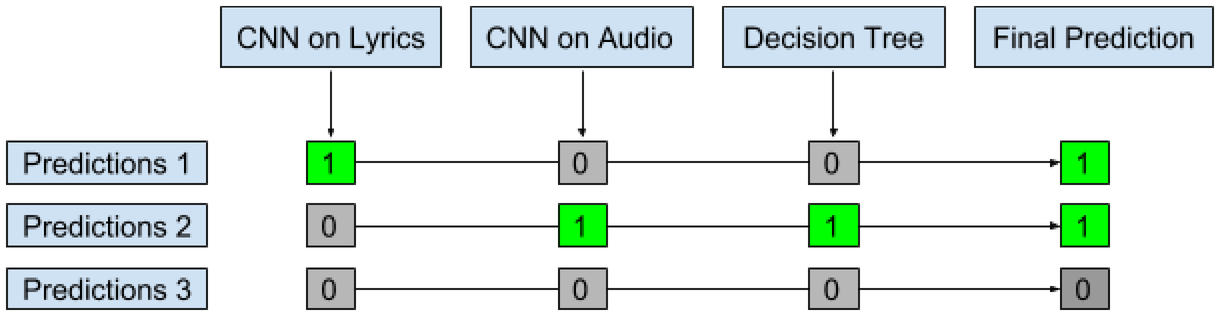


Figure 15: Optimistic Stacking

Examples of false positive

We looked at examples of false positives, where our model predicted a hit yet the song didn't achieve the maximum popularity above 70 in the dataset. An interesting example are five songs by Chris Brown. While the songs individually didn't achieve the predefined popularity threshold, the album "Heartbreak on a Full Moon" they are featured in made it to the 28th place on US Billboard 200 in 2018. This supports the potential of our model to discover music that is not a hit yet, but which may still continue growing in popularity.

Conclusions and future work

Using more features, audio, lyrics, artists, producing teams, etc., we were able to achieve 40% recall, or predict 4 true hit songs out of every 10 predicted hit songs, which was a huge improvement compared to out baseline model that only achieved 5% recall (0.5 true hit songs out of every 10 predicted hit songs).

This achievement is meaningful for Warner Music Group. According to the project mentor, Ryan Faus, the marketing team at Warner Music Group have to listen to songs from a long list one by one in order to decide which song to promote, or which artist they want to sign. The whole process is really inefficient, but our model could help narrow down their "candidates list" to speed up and optimize the selection process.

For future work, we would like to implement the CNN model using lyrics with the Siamese CNN and only use one neural network to predict using both audio and lyrics, as there may be some interesting relationships between the two that we couldn't capture in the separate models.

Acknowledgments

The authors thank Ryan Faus for providing data sources and advice and thank Warner Music staff for technical assistance. We also thank Professor Brian McFee and Professor Sam Bowman for their advice on the models.

Bibliography

[1] Topic Modeling in Python with Gensim. (2018, December 04). Retrieved from https://www.machinelearningplus.com/nlp/topic-modeling-gensim-python/

[2] Yu, L., Yang, Y., Hung, Y., & Chen, Y. (n.d.). HIT SONG PREDICTION FOR POP MUSIC BY SIAMESE CNN WITH RANKING LOSS.

[3] Askin, N., & Mauskapf, M. (2017). What Makes Popular Culture Popular? Product Features and Optimal Differentiation in Music. American Sociological Review, 82(5), 910-944. doi:10.1177/0003122417728662

[4]Lang-Chi Yu, Yi-Hsuan Yang, Yun-Ning Hung, Yi-An Chen. HIT SONG PREDICTION FOR POP MUSIC BY SIAMESE CNN WITH RANKING LOSS.