Uncovering What Makes A Hit Song

Hao Ruoxin[1] **Lim Chun Han**[2] **Liu Xinyu**[3] **Ural Malik**[4] **Yan Jinjin**[5]
https://github.com/limchunhan97/bt5153_group_project

Abstract

We develop a model that can predict the popularity of a song on the popular music streaming platform Spotify.

1. Introduction

Music has for a long time been a source of entertainment for many people. Naturally, there will be some songs that are more popular than others, with some songs having an enduring popularity even decades later such as the classics from The Beatles and Michael Jackson. On the other hand, there are also songs that never ever gain any traction and remain unknown to the public. We are interested to understand the features that make a song popular and develop a model that can predict the popularity of a song on the popular music streaming platform Spotify.

There are mixed opinions over whether there is a fixed pattern to the factors that make a popular song. In a study conducted by the Columbia Business School, Mauskapf and Askin analysed 60 years' worth of tracks from the Billboard Hot 100 and concluded that songs that top the charts tend to be different from the predecessors, although the extent of difference allowed has an upper bound (Morris n.d.). Blume supports this, saying that even though most new songs will incorporate some features from songs that were previously successful, no two songs will be exactly the same (Blume 2019). Songs that want to make an impact must have something fresh and new. On the other hand, Emamzadeh presents a more nuanced opinion (Emamzadeh 2018). He opines that songs need to be similar enough to evoke a sense of familiarity but different enough to be successful. But what is similar enough? Is there a pattern to this similarity? These are questions we hope to answer in our project.

There are a few factors that can determine the popularity of a song. Blume argues that the lyrics and melody of a song are two key factors (Blume 2019). Lyrics have the ability to provide fresh ways of expressing the same ideas, while an unforgettable melody can enable a song to be etched in a person's memory. For these reasons, these will be two key factors we will be looking at closely in order to uncover the underlying patterns behind popular and unpopular songs. Our own experience also indicates that the popularity of a song may be closely tied to the popularity of the artists. An artist who is already more popular is intuitively more likely to have more popular songs. For the completeness of our analysis, we will also be evaluating the impact that an artist's popularity has on the popularity of a song.

Being able to produce a popular song can be a massive fate changer. For new and upcoming artists, a popular song can thrust them into the center of the world stage and kickstart their career. For existing heavyweights in the industry, failure to do so can mean the end of their careers. We hope to be able to develop a model to help record labels produce songs that resonate with the crowd and continue to touch the hearts of millions around the world one song at a time.

2. Motivation

We have identified the following 3 hypotheses that we plan to test during our project:

2.1 Popularity of a song is determined by its various features

A song can be considered as a combination of different components such as its audio features (loudness, tempo, etc.), lyrics, and artist details. To analyze the popularity of a song, we will be analyzing the data for each of these components to find meaningful insights and trends. We hypothesize that the data corresponding to the song may be able to help us predict the popularity rating of a song according to the current popularity rating system (as determined by Spotify).

2.2 Lyrics play a very important role in building connections with listeners

Music resonates with listeners not only through the melody, but also through lyrics. We will analyze the language complexity, emotional tendency and topics of lyrics to provide insights into the kinds of lyrics that will resonate

[1]A0218868L [2]A0218837U [3]A0218834Y [4]A0218898E [5]A0218863W

more strongly with listeners of the current period. We will be looking at the lyrics of popular songs, which we define as songs that appear on the Billboard 200 dataset between 1963 and 2019.

2.3 Changes in features of popular songs over time

We hypothesize that the features of popular songs (such as audio features, language complexity, explicit content, and popular topics) evolve over the years. Through this, we aim to investigate how musical tastes have changed over time.

3. Methodology

The overall project methodology followed 4 key steps:

3.1 Data Creation and Preprocessing

After developing the overall project objective, we started the data collection process. We created the dataset by integrating data collected from the following key sources: Spotify, Genius and Billboard 200. The Spotify dataset provided the details about artists, song names, and various audio features. The Genius dataset was utilized to obtain the lyrics data for the songs in our dataset. Finally, the Billboard 200 dataset was utilized in identifying the song that were popular in the different time periods over the past few decades.

3.2 Hypothesis Creation

As we wanted to analyze the impact of music from multiple angles, we thought of various hypotheses that would be meaningful to validate through the study. We shortlisted 3 key hypotheses that were most relevant and would be suitable to be analyzed through the dataset that we had created. However, over the course of the study we realized we wanted to expand the scope of the study, and hence incorporated data from additional sources such as Genius and Billboard 200.

3.3 Model Building and Results

To predict the popularity rating of the songs, we utilized various machine learning algorithms such as SVM, XGBoost, and Neural Networks. We chose the top 3 models and then performed grid search to identify the best hyperparameters. The model evaluation metrics chosen were RMSE and MAE.

3.4 Model Building and Results

To validate the 3 hypotheses shortlisted before, we utilized the results from the models. We identified the relative importance of the various features and their impact on the popularity of a song. We also conducted a temporal analysis of the features to identify the key trends and changes in the various features.

4. Methodology

4.1 Data Collection & Aggregation

In the preparation of our dataset, we have obtained data from three main sources. The first is a dataset consisting of all the songs in Spotify scraped using the Spotipy API, which is curated by Yamac Eren Ay (Ay 2021). This comprehensive dataset contains important information of each song, such as the release date, genres and audio features that quantify the musical characteristics of a song. Additionally, we also used the Spotipy API ourselves to scrape for the number of followers for all Spotify artists that are mentioned in our Spotify dataset.

The second source is a dataset we found online created by Andrew Thompson that contains the acoustic features of all songs on the Billboard 200 from 1963 to 2019 (Thompson 2019). We will be joining this with the first dataset containing songs from Spotify and tagging songs in the Spotify dataset that also appear in this dataset with a value of 1 under the dummy variable ‘popularity_dummy’.

The third source is LyricsGenius, a Python client for the Genius.com API, which we used to obtain the lyrics of the songs we have in the Spotify dataset. We will be using the song and artist name as the parameters to request the lyrics from the API.

Our combined dataset has 163,712 rows and 22 variables. See Table 1 for full list of variables. Description of some acoustic features are as provided by Morris in his study of 60 years’ worth of tracks from the Billboard Hot 100 (Morris n.d.).

4.2 Data Preprocessing

4.2.1 Measuring popularity

In our datasets, we have two measures of popularity. The variable ‘popularity’ measures the song’s popularity on Spotify currently, while the variable ‘popularity_dummy’ indicates that a song was ever listed on the Billboard 200. This is so that we can conduct our analysis in two dimensions – one analysing a song’s current popularity, and the other analysing a song’s popularity at its peak. As such, it is possible that a song may have a low score in ‘popularity’ but have a value of 1 for ‘popularity_dummy’ because it is an old song and was popular years ago.

4.2.2 Joining Spotify and Billboard 200 Dataset

To join the first and second datasets, we first dropped duplicates in the respective datasets to prevent conflicting information from being present in the dataset. As a consequence, we ensured that every row contained a

unique song in terms of acoustic features, song name, artist name and song duration. For duplicates, the most popular one was kept while the rest were removed from the dataset. In other words, covers or remastered versions of the same song by the same artist were included as long as the audio features were uniquely different. We then joined the two datasets based on their Spotify ID, which was a variable present in both datasets.

4.2.3 Joining With Artist Popularity

We joined the main Spotify dataset with artist popularity dataset obtained from Spotipy API, using the following steps.

First, pick the first artist name from the artist name feature in the main dataset. This is to account for cases where there are multiple artists for a song. Second, left join main dataset with artist popularity dataset on the artist name (case-insensitive). For 3,172 rows that have no matches, further relax the criteria by picking the text before any punctuations since that some names have suffix in one of the datasets. At the same time, after applying the more relaxed criteria, the length of the name should be longer than 3 to avoid multi-match problems caused by non-English letters. See the example in Appendix Figure 1.

Then, join again, and the number of null values decreases to 2,973. At last, fill all the remaining null values with the average artist popularity.

4.2.4 Removing Incomplete Data

Combining the dataset from different resources and doing preliminary cleaning gives our dataset 163712 records. However, due to the limitations of the LyricsGenius API, we were not able to get the lyrics for some songs. So, we checked the distribution of the whole dataset and compared it against a subset of the dataset containing only the songs with lyrics. We found that the distributions of the popularity score and features were similar across both datasets. This gave us confidence to remove the songs without lyrics as existing patterns in the dataset would not be removed, reducing our dataset to 93627 records. Subsequently, as we found that there were some non-English songs in the dataset, we used LangDetect to detect the language of lyrics and compared the similarity of distribution of popularity scores and features between English songs and non-English songs. Similarly, we found that the distribution is similar, giving us confidence to remove non-English songs as well. The comparison of distributions is shown in Figure 1. By removing non-English songs, our final dataset was further reduced to 83,074 songs.

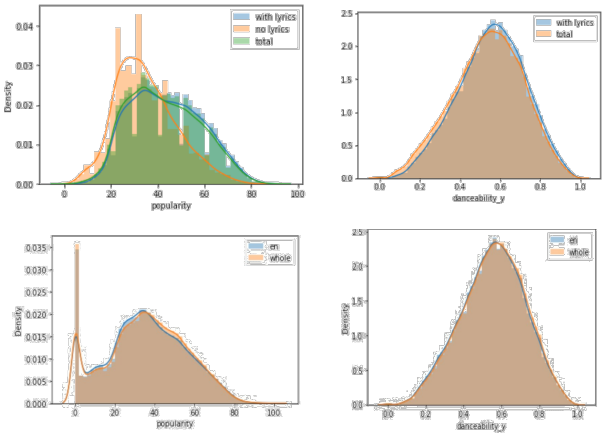

Figure 1. Popularity Distribution of samples and whole dataset

To further clean the data, we also identified outliers that met the condition where the value for a particular feature was higher than Q3+1.5* IQR or lower than Q1-1.5*IQ. This was set according to our domain knowledge. By detecting outliers using this method as shown in Figure 2, we found there are some audio books which have sentences and words that are too long sentences. In addition, some lyrics scraped using the LyricsGenius API also provided incomplete lyrics. We decided to remove these outliers as our objective was to predict the popularity scores of songs and removing these outliers would make our analysis more meaningful. After carrying out these steps to further clean our dataset, we obtained a total 66,452 records for our final dataset.

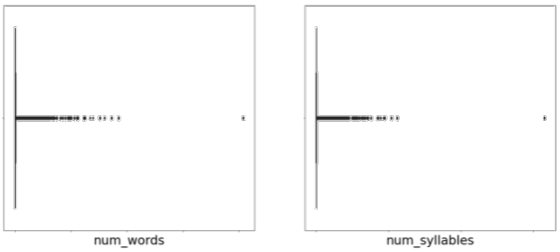

Figure 2. Boxplot of num_words and num_syllables

4.2.5 Cleansing Lyrics

The cleansing process contains several steps before we could proceed into feature engineering and LDA modelling.

- **Basic Cleansing**, including removing URLs, html tags, emojis, translating Chinese punctuations into English versions, removing parentheses and the content in between (like, '[Verse 1]'), converting characters to lowercase, removing standalone alphabets and necessary punctuations.
- **Deep Cleansing** (for LDA topic modelling): including strictly removing punctuations and stop-words, removing all non-alphanumeric characters and lemmatizing

4.2.6 Data Normalization & One-Hot Encoding

Since we need to develop multiple models for further popularity prediction, we normalized the dataset to cater for the need of scale-sensitive models like linear regression and SVM. Specifically, we used the standard scaler to normalize our data.

Categorical features were also one-hot encoded to convert them into multiple dummy variables.

4.3 Sentiment Analysis

4.3.1 Basic Ternary Sentiment

Sentiment analysis is used to extract opinions from song lyrics as positive, negative, or neutral. We utilized Natural Language Toolkit (NLTK's VADER module) to obtain the sentiment scores of the lyrics.

VADER is a rule-based sentiment analyzer. It uses lists of lexical features (such as words) that are marked as positive or negative according to their semantic direction to calculate the textual emotion, which returns a dictionary and allows for the calculation of the probability of the sentence to be positive, negative, or neutral. The polarity scores range from -1 (negative) to 1 (positive). The sentence will then be assigned a sentiment in which it has the highest probability, giving the compound score.

4.3.2 BalanceNet

Taking reference from Timothy Liu (Liu n.d.), we developed this BalanceNet for our mixed neural network consisting of both LSTM and CNN since either of the model will learn a certain pattern of the texts input and combination of two will unleash the full power of both models. RNNs, especially LSTMs, are good at learning the significance of the order of sequential data like texts and time-series while CNNs are capable of extracting features from data to identify them.

Instead of solely using either of the two models, we take elements from each of the abovementioned models and create a multi-channel neural network structure as shown above where we permit the model itself to decide which channel to take to get more accurate predictions. We hypothesize that this will allow the model to take advantages of both models to make overall better predictions.

Based on BalanceNet structure, we can classify our lyrics into sentiment classes beyond binary (positive/negative) or ternary (positive/negative/neutral) classes. This new approach of multinomial classification of sentiments may improve our classification accuracy, which may then help in our regression analysis of popularity scores. To be specific, we will classify text emotions into 5 categories: *sad, neutral, happy, anger, hatred* respectively.

Getting a sentiment score by analyzing lyrics is also a very important step for our feature engineering. We can use sentiment scores or the multi-label dummy variables as an additional set of features in predicting the popularity of a song. In addition, we can also compare the difference between popular and unpopular songs and see whether the sentiment scores of popular songs change over time.

4.4 Language Complexity Analysis

Intuitively, pop songs are often filled with simple and easy words since intricate terminologies would find it more difficult to build rapport with audiences and gain traction. Considering this, we calculated language complexity scores of the lyrics which will serve as additional features of our prediction model to evaluate the influence of lyrics on a song's popularity.

4.4.1 Language complexity

Language complexity will be measured from two angles: how readable the text is (textual readability) and how rich it is (textual richness) (Ballandonne and Cersosimo 2020).

4.4.2 Lexical readability

Lexical readability will be assessed using Flesh-Kincaid readability score, which is based on word length and sentence length.

$$FKScore = 206.835 - 1.015\left(\frac{\text{total words}}{\text{total sentences}}\right) - 84.6\left(\frac{\text{total syllables}}{\text{total words}}\right)$$

Type-Token Ratio (TTR) or Hapax richness will be used to evaluate textual richness. The basic idea behind this is that if the text is more complex, it uses a more varied vocabulary and hence has a larger number of unique words.

$$Type - Token\ Ratio = \frac{\text{total number of unique words}}{\text{total words}}$$

$$Hapax\ richness = \frac{\text{the number of words that occur only once}}{\text{total words}}$$

5. Investigating Hypotheses 1 and 2 – Predicting a song's popularity

5.1 Exploratory Data Analysis (EDA)

Figure 3 shows the distribution of Spotify popularity score after all the data cleansing, which nearly follows a normal distribution with an average of 40. Then, we looked at the correlation between the target variable of the Spotify popularity score and the explanatory variables. As we can see from Figure 3, artist popularity shows a strong positive correlation with Spotify popularity. In addition, acousticness, loudness and number of words have moderate correlations with the popularity score.

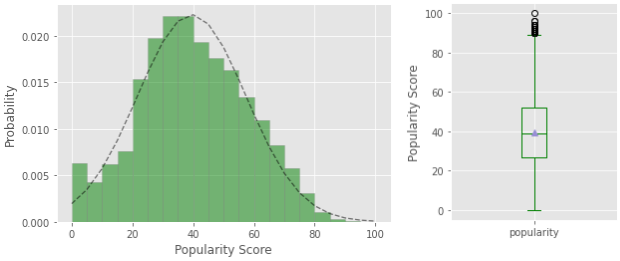

Figure 3. Distribution of Spotify popularity score

Appendix Figure 1 shows the scaled audio features of the top 3 songs based on their popularity score on Spotify. We can see that the distribution of audio features is very different among the three songs, which is consistent with the notion established by studies we have found that there is no secret recipe to producing a great song. However, one thing all three songs have in common is that they all have a special audio feature that is particularly prominent as we can see in Appendix Figure 1, although this audio feature may differ among songs. With this in mind, we will investigate what some of the important attributes that make a song popular are and analyze how significant they are in influencing a song's popularity. In addition, we will also consider how the lyrics of a song may affect the song's popularity when developing our prediction model.

5.2 Prediction Modelling

For this project, our main objective was to give an accurate prediction of the popularity score of a song and identify the factors that may influence its popularity. To test the performance of our model, we first split the dataset, with 80% of the dataset assigned to be the training dataset and the remaining as the holdout test set.

The final list of features used in our modelling is as described in Table 1. (See Appendix Table 1 for more details)

Table 1. List of Variables

Variable Categories	*Variables*
Target Variables	Popularity
Basic Features	Id, Song, Album, Artist, Duration_ms, Key, Mode, Time_signature, Album_id, Date, Date_year, Duration_ms_d, Lyrics, Artist_popularity
Acoustic Features	Acousticness, Danceability, Energy, Instrumentalness, Liveness, Loudness, Speechiness, Tempo, Valence
Generated Features	Num_sentences, Num_words, Num_syllables, Readability, Grade, Richness, Language, Popularity_dummy
Sentiment Features	Negative, Neutral, Positive, Compound, Compound_vader_polarity, Is_happy, Is_neutral, Is_sad, Is_hatred, Is_angry

For our modelling, we first used the linear regression model as the baseline model. We then selected the SVM, decision tree, random forest, XGBoost, Adaboost and neural network models to compare and evaluate the performance of the different models. The advantage of using linear regression is that it is easy to implement and is a simple model to interpret important features. From the heatmap in Appendix Figure 1, we can also see that there are some features have a significant linear correlation with the popularity score of the song such as artist popularity, acousticness and loudness, thus suggesting that a generalised linear model may work well. We also decided to try tree-based models due to their ability to model non-linear relationships, which has often led to higher prediction accuracy and robustness. We also decided to try using a neural network model as it could potentially outperform the other models since it uses a sophisticated architecture with designed activation function and can model complex relationships between our features and the target variable.

Table 2. Model Performance

Model	*MAE*	*RMSE*
Linear Regression	*11.45*	*14.26*
Random Forest	*10.20*	*13.00*
Decision Tree	*14.36*	*18.69*
SVR	*10.58*	*13.47*
XGBoost	*10.21*	*13.00*
AdaBoost	*12.38*	*15.02*
Neural Network	*10.44*	*13.24*

Using MAE and RMSE as our evaluation metrics, we find that random forest, XGBoost and neural network models have relatively better performance. Hence, we select these three models to do further hyperparameter tuning.

5.3 Hyperparameter Tuning

We conducted a grid search with a 5-fold cross validation to identify the best hyperparameters that would provide the lowest RMSE scores possible for our 3 best models: Random Forest, XGBoost and Neural Network. Table 3 shows the final grid search results.

Table 3. Model Performance After Hyperparameter Tuning

Model	*Parameter*	*MAE*	*RMSE*
RF_Best	*max_depth=20, n_estimators=800, min_samples_split=0.0001*	*10.15*	*12.92*
XGBoost _Best	*learning_rate=0.01, max_depth=10, n_estimators=800*	*10.00*	*12.75*
Neural Network_ _Best	*activation='relu', 'linear' optimizer='adam', batch_size=64, epochs=14, lr=0.001*	*10.35*	*13.20*
XGBOOST_BEST	*WITH 10 MOST IMPORTANT FEATURES GIVEN BY PFI AND SHAPLEY*	*10.16*	*12.93*
XGBOOST_BEST	*WITH 10 MOST IMPORTANT FEATURES GIVEN BY XGBOOST*	*10.21*	*12.98*

For the random forest model, the hyperparameters and corresponding range of values tuned are as follows: n_estimators [200, 400, 600, 800], max_depth [20, 40, 60, 80] and min_samples_split [0.0001, 0.0005, 0.001, 0.005].

The optimal combination found using grid search was n_estimators = 800, max_depth = 20 and min_sampless_split = 0.0001, all of which are on the upper limit of range of values tested. Therefore, we took a deeper look at grid search mean test score of each combination.

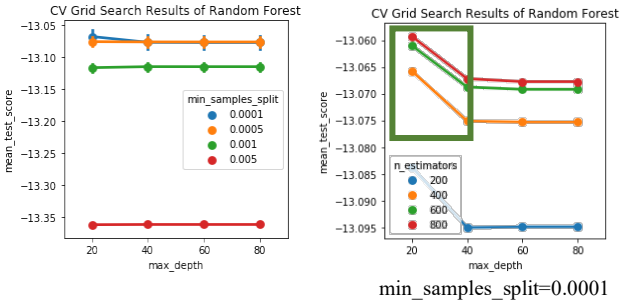

Figure 4. GridSearch Result for Random Forest

As we can see from Figure 4, the RMSE of min_samples_split = 0.0001 and min_samples_split = 0.0005 min_samples_are very close, and when this parameter becomes a very small number, there is a risk that the random forest model might be overfit. Hence, we decided to stop at 0.0001. Likewise, when we increase n_estimators from 600 to 800, the improvement in model performance is very little. As such, to preserve reasonable runtime and prevent overfitting, we decided to stop at 800. In addition, given that the RMSE decreases when the max_depth is reduced from 40 to 20, we decided to also try lower values of max_depth but found that the optimum value is still 20.

When tuning the hyperparameters for our XGBoost model, we followed the same approach as that for our random forest model. The search space consisted of the following hyperparameters and range of values: learning_rate [0.001, 0.01, 0.1, 0.2], n_estimators [200, 400, 600, 800] and max_depth [20, 40, 60, 80]. The optimal set of hyperparameters found using grid search was learning_rate = 0.01, n_estimators = 800 and max_depth = 20, among which both n_estimators and max_depth is on the upper limit of the range of values. Thus, we also looked at the mean test score for each combination as shown in Figure 5.

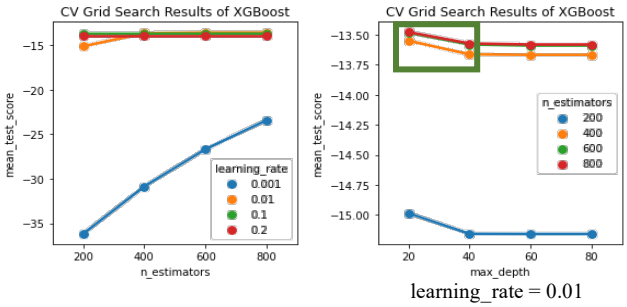

Figure 5. GridSearch Result for XGBoost

Gradient boosting models learn quickly and could easily overfit training data (Brownlee 2020). Thus, we set the learning_rate as 0.01 to slow down the corrections by new trees when added to the model. Similarly, considering both performance improvement and computational costs, we did not increase the number of estimators to be beyond 800 and decided to tune the max_depth instead. In conclusion, the best set of hyperparameters for the XGBoost model is learning_rate = 0.01, n_estimators = 800 and max_depth = 10. This model resulted in a MAE of 10 and RMSE of 12.75, which is the best among all our models.

Finally, we tuned the neural network model. The ReLU activation function is often an effective activation function for regression neural network models and as such was the activation function of choice for our layers. Similarly, as we are dealing with a regression problem, we used the mean squared error as our loss function. Our choice of optimizer as "Adam" as it combines both gradient descent with momentum and the root mean squared propagation algorithms (Geeksforgeeks 2020). When constructing a neural network model, there exists a trade-off between the number of epochs used to the train the model and the batch size used for gradient descent in terms of model performance and runtime. One can either increase the batch size to have less iterations per epoch or reduce the batch size which means more iterations and updates are required per epoch. As such, we set the search space for the following hyperparameters and corresponding range of values: batch size [32, 64, 128, 256] and epochs [10,11,12,13,14,15,16]. After training the model on two

sets of features (one with sentiment categories obtained using VADER and another with sentiment categories obtained using BalanceNet), we found that the best set of hyperparameters which gave the lowest MSE was 64 for batch size and 14 for epochs. This resulted in a better model performance for the neural network model. In the future, we can try to expand the grid search parameters to include tuning the number of layers, dropout rate and using a different optimizer.

5.4 Machine Learning Interpretability

We further interpret the feature importance of the best XGBoost model both globally and locally.

5.4.1 Global Interpretability

We first looked at the model's global feature importance using Permutation Feature Importance (PFI) and Shapley Values and compared them with XGBoost's feature importance scores, as seen in Figure 6 and Appendix Figure 4 & 5 in appendix. The 10 most important features given by PFI and Shapley are similar. The only difference between XGBoost's feature importance and that of Shapley and PFI is that instead of danceability, instrumentalness is on the top 10 list. This suggests that regardless of the method used, global feature importance is likely to be robust as similar features are regard to be important across different methods.

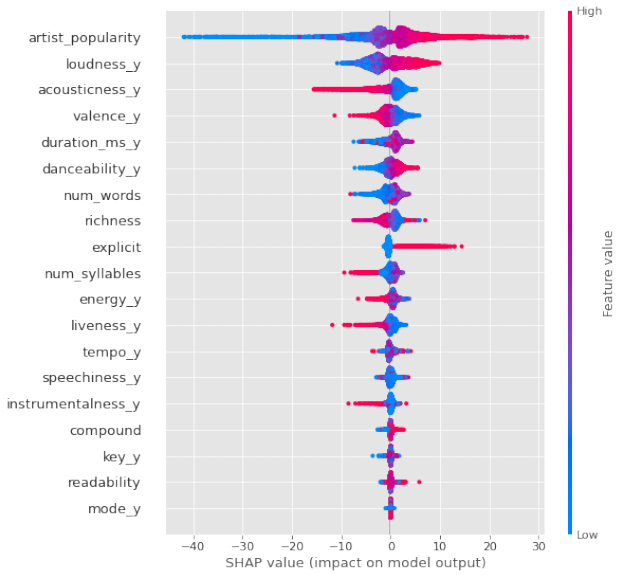

Figure 6. SHAP value of XGBoost Model

By observing the feature importance obtained from the different methods, we found that a few attributes that most methods agree are important are artist_popularity, loudness, explicit, and duration. From their Shapley values as seen in Figure 6, we can see that for artist_popularity and loudness, a higher feature value results in a higher popularity score while for acousticness and valence, a negative feature value results in a lower popularity score.

5.4.2 Local Interpretability

Next, we had a closer look at songs with the highest and lowest predicted popularity scores. We found that our prediction popularity scores were actually very close to the actual popularity score for both songs. With this, we then looked at local feature importance for these two songs using Shapley local values and LIME, as shown in Appendix Figure 6 & 7.

The local feature importance given by Shapley and LIME are similar. Comparing the two cases, artist_popularity plays a very important role in predicting popularity score. By Yourself has the highest prediction score because it is by more popular artists and has explicit lyrics, higher danceability and loudness. However, the liveness of the song negatively affects its popularity score. The Purge of History has the lowest prediction score due to it being by less popular artists, having too complex lyrics (higher num_syllables, richness), and having a shorter duration. However, the danceability of the song contributes positively to its popularity score.

5.5 Feature Selection

To try further improving our model performance, we selected the 10 most important features given by XGBoost's feature importance, PFI and Shapley and refit the best XGBoost model. Table 3 lists the results. The model performance does not improve but is still quite good. The model with the 10 features given by PFI and Shapley performs slightly better than the one with the 10 features given by XGBoost's feature importance.

5.6 Looking at the topics of lyrics

To understand the topics of the lyrics of popular songs and unpopular songs, we made use of Latent Dirichlet Allocation (LDA) models to cluster the words found in the lyrics into topics.

LDA classifies documents into topics by allocating topics to each document model and allocating words to each topic model. Each preprocessed lyric will be modelled as a multinomial distribution of topics, and each topic is modelled as a multinomial distribution of keywords. By briefly describing the documents and retaining the essential feature information, the LDA helps to efficiently process large-scale data sets into clusters of topics (Mutanga and Abayomi 2020).

Popular songs refer to those that appeared on the Billboard 200, as identified by a dummy variable that we have

assigned for this. To enable the topics to be more interpretable, we only kept parts of the lyrics that were nouns. These words were identified by using part-of-speech tagging.

The number of topics were chosen based on the coherence and perplexity scores that the resulting number of topics gave. Coherence score measures the degree of semantic similarity between high scoring words in a topic2 while perplexity score measures how well the model performs on new data (Kapadia 2019). In our LDA models, we used a chunk size of 30000 and 10 passes for each iteration. The corpus and dictionary used were based on the lyrics of the relevant songs. We tried between 1 and 10 topics and chose the model that had the best trade-off between coherence and perplexity scores.

Subsequently, we used the pyLDAvis package to visualize the topics and adjusted the relevance metric to extract the most important terms for each topic. We decided on the topic label based on the keywords found.

Figure 4. Wordcloud plot of popular songs topic1

For the popular songs, we found that they were generally about either street life, romantic love or Christmas. For unpopular songs, we found that they were generally about daily life, street life and Christianity.

From this analysis, we can see that we cannot make a conclusive statement that a particular topic will make a song more popular as songs about street life can be either popular or unpopular. However, it is observed that songs about love tend to be popular songs.

6. Investigating Hypotheses 3 - A temporal analysis of song features

In this section, we wanted to explore how acoustic and lyric features of popular songs changed over time and investigate if there has indeed been a change.

6.1 Acoustic features

From 1963 till now, we found that over time, danceability and loudness has increased for popular songs, as shown in Figure 8. Meanwhile, valence, which describes the musical positiveness conveyed by a track, decreases. In the last 10 years, it is clearly shown that duration of songs has become shorter.

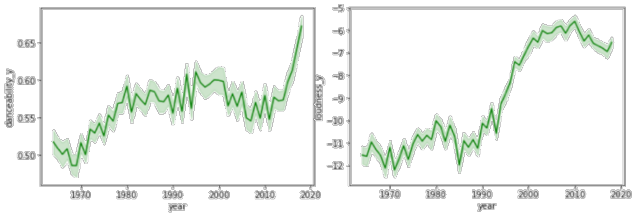

Figure 8. Trend of danceability and loudness

6.2 Lyric features

By observing how lyrics features have changed, we found that artists of popular songs today tend to write longer songs than popular songs of the past as their lyrics have a greater number of words and number of syllables.

Over the years, we also observe that the readability score of lyrics in popular songs have decreased. Readability score is based on the number of words, number of sentences and number of syllables in the lyrics. A lower readability score means that the lyrics of songs are harder to understand.

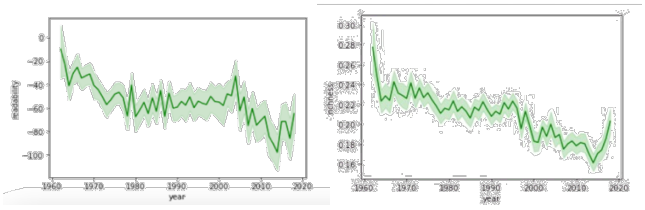

Figure 9. Trend of readability and richness

We also found that the richness in the lyrics of popular songs have decreased. Richness refers to the number of unique words used in the lyrics. This suggests that lyrics in popular songs in recent times use fewer unique words and are less complex. Figure 9 shows the trend about readability and richness.

Further, in Figure 10, we found that for popular songs, most of them delivered happy emotions through their lyrics. Only a small proportion of songs have lyrics with angry emotions. We also found that throughout the time period observed, the proportion of songs with lyrics for each type of emotions observed remained the same.

We also found that the number of popular songs which have explicit lyrics have been increasing over the years with a steep increase from 2010, which can be seen in Appendix Figure 11 below.

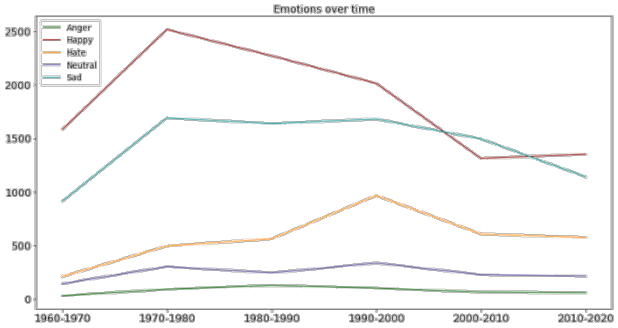

Figure 10. Trend of number of songs in 5 emotions

Finally, the compound score, which indicates the overall sentiment of songs, has decreased over time for popular songs, suggesting that the lyrics of songs have less positive sentiment on average. However, the average scores indicate that the lyrics of popular songs today still have an overall positive sentiment, which is shown in Appendix Figure 12

6.3 Topic of Lyrics

To analyze how the topics of lyrics have changed over time, we first divided the songs into different bins based on the decade that they were released in. Lyrics of popular songs released in each decade were then analyzed to derive key topics.

Songs on the Billboard 200 that were released between 1930 and 1960 were not considered due to their relatively small sample size. This was because our Billboard 200 dataset only covers songs that were on the Billboard 200 between 1963 and 2019. It is possible for songs that were released before 1963 to still be in the dataset if it continued to be popular in 1963 and beyond. However, as one would expect, such songs are in small numbers.

Similarly, we used the method described in Section 5.6 to identify key topics for the lyrics of songs in each decade. Number of topics were decided based on the coherence and perplexity scores of the model, and we also adjusted the relevance metric to focus on the most important terms of each topic.

Table 4. Topics of Lyrics by Decades

Generation	*Topics*
1960 - 1970	*Romantic love, adventure, daily life*
1970 – 1980	*Positivity, family*
1980 – 1990	*Romantic love, street life, party*
1990 – 2000	*Romantic love, street life*
2000 – 2010	*Heartbreak, street life, party, desperation*
2010 – now	*Romantic love, street life*

The topics found for each decade of songs is as seen in the table above. From this, we can see that romantic love seems to be a common topic for popular songs across different decades. Songs about street life involving the use of several profanities also became popular from the 1980s onwards.

7. Conclusion

With our model, we were able to predict a song's popularity score on Spotify with a RMSE of 12.75, which is good considering that the popularity score ranges from 0 to 100. By leveraging on machine learning interpretability methods, we found that the artist's popularity, loudness of the song, acousticness of the song and duration of the song are among some of the most important features in determining a song's popularity. For a record label, this means that if they want to produce a popular song, they should find an artist with good social media presence, make songs louder, steer away from acoustic songs and make longer songs.

Through our temporal analysis of popular songs, we also found that the features of popular songs have indeed changed over time. In terms of acoustic features, popular songs have higher danceability and loudness and the valence of songs has decreased. In terms of lyrical features, songs nowadays have more words and more syllables. Songs also have a lower readability, a smaller number of unique words and less complexity. In terms of the sentiment of lyrics, most popular songs of each decade are still about happy emotions, although the overall sentiment of songs have become less positive. There have also been more popular songs with explicit lyrics, with a steep increase from 2010. In terms of topics, there are more songs about street life in recent years and songs about romantic love seem to be a consistent theme over many decades.

From these analyses, we can see that it is not necessarily the case that all popular songs are different and that there is no secret recipe to a popular song as our model has identified patterns that define a popular song in today's terms. However, it is also important to note that we also found out that what makes a popular song changes over time. While our model may focus on understanding what defines the popularity of a song in today's terms, our temporal analysis also showed that features of popular songs have changed over time. This makes the definition of a popular song everchanging. Therefore, constant analysis of what makes a song popular needs to be done in order to fully understand the underlying trends that drive the music industry today.

References

Ay, Y. (2021, April 20). Spotify dataset 1922-2021, ~600k TRACKS. Retrieved April 25, 2021, from https://www.kaggle.com/yamaerenay/spotify-dataset-19212020-160k-tracks

Ballandonne, M., & Cersosimo, I. (2020). Toward a "text as DATA" approach in the history and methodology of economics: An application to ADAM Smith's Classics. SSRN Electronic Journal. doi:10.2139/ssrn.3595120

Blume, J. (2019, January 23). What makes a song a hit? Retrieved April 25, 2021, from https://www.bmi.com/news/entry/what-makes-a-song-a-hit

Bouazizi, M., & Ohtsuki, T. (2016). Sentiment analysis: From binary to multi-class classification: A pattern-based approach for multi-class sentiment analysis in twitter. *2016 IEEE International Conference on Communications (ICC)*. doi:10.1109/icc.2016.7511392

Brownlee, J. (2020, August 27). Tune learning rate for gradient boosting with xgboost in Python. Retrieved April 25, 2021, from https://machinelearningmastery.com/tune-learning-rate-for-gradient-boosting-with-xgboost-in-python/

Components. (n.d.). Acoustic and meta features of albums and songs on the Billboard 200 (large). Retrieved April 25, 2021, from https://components.one/datasets/billboard-200-with-segments

Gao, M., Li, T., & Huang, P. (2019). Text classification research based on Improved word2vec and CNN. *Lecture Notes in Computer Science,* 126-135. doi:10.1007/978-3-030-17642-6_11

Intuition of Adam Optimizer. (2020, October 24). Retrieved April 25, 2021, from https://www.geeksforgeeks.org/intuition-of-adam-optimizer/

Kapadia, S. (2019, August 19). Evaluate Topic Models: Latent Dirichlet Allocation (LDA). Retrieved April 25, 2021, from https://towardsdatascience.com/evaluate-topic-model-in-python-latent-dirichlet-allocation-lda-7d57484bb5d0

Liu, T. (n.d.). Tlkh/text-emotion-classification. Retrieved April 25, 2021, from https://github.com/tlkh/text-emotion-classification

Mutanga, M. B., & Abayomi, A. (2020). Tweeting on COVID-19 pandemic in South Africa: LDA-based topic modelling approach. *African Journal of Science, Technology, Innovation and Development,* 1-10. doi:10.1080/20421338.2020.1817262

What makes a HIT: 60 years of #1 songs. (n.d.). Retrieved April 25, 2021, from https://www8.gsb.columbia.edu/articles/projects/what-makes-a-hit/

Why do some songs become popular? | Psychology Today. (n.d.). Retrieved April 25, 2021, from https://www.psychologytoday.com/intl/blog/finding-new-home/201806/why-do-some-songs-become-popular

Appendix

Main Spotify Dataset

	id	...	artists_d	spotify_pop	artist_fir	artist_fir_clean	artist_name	artist_popularity	name_fir_clean
738	7JKQQgFNw5RXiJBuLf7dXe	...	Hank Williams	46	hank williams	hank williams	Hank Williams, Jr.	69	hank williams
7155	6XNim2ZrMOMpO3RQIOAcFh	...	D.Y.	0	d.y.	d	D'Jamency	0	d
7156	6XNim2ZrMOMpO3RQIOAcFh	...	D.Y.	0	d.y.	d	D.Semsis	0	d
7157	6XNim2ZrMOMpO3RQIOAcFh	...	D.Y.	0	d.y.	d	D.roitman	0	d
7158	6XNim2ZrMOMpO3RQIOAcFh	...	D.Y.	0	d.y.	d	D.Hemalatha Devi	0	d

Artist Popularity Dataset

Figure 1. Example of Joining Artist Info to Main Dataset

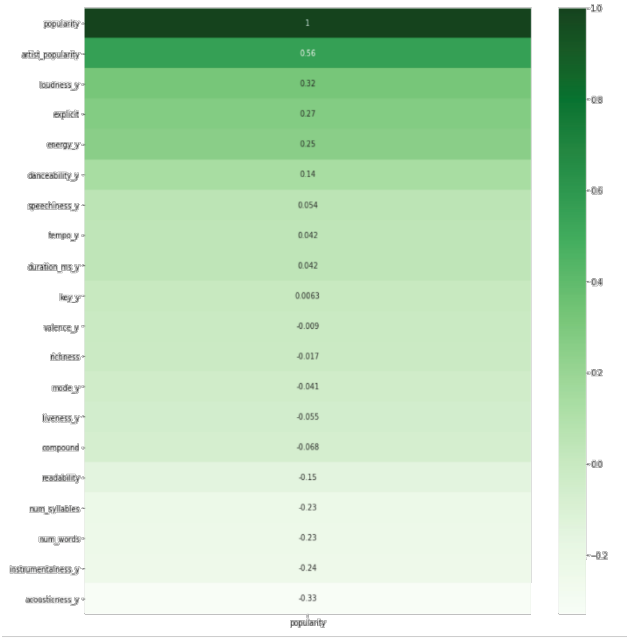

Figure 2. Heatmap of feature correlation with popularity score

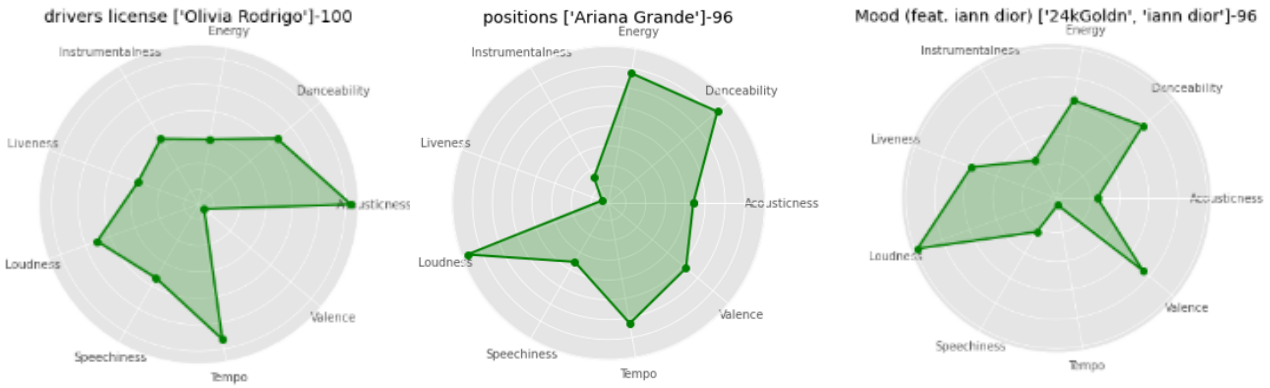

Figure 3. Radar map for top3 songs

Weight	Feature
0.3185 ± 0.0072	artist_popularity
0.1346 ± 0.0087	loudness_y
0.0618 ± 0.0069	acousticness_y
0.0373 ± 0.0042	duration_ms_y
0.0367 ± 0.0019	valence_y
0.0363 ± 0.0051	explicit
0.0235 ± 0.0015	danceability_y
0.0232 ± 0.0017	num_words
0.0217 ± 0.0018	richness
0.0186 ± 0.0023	num_syllables
0.0159 ± 0.0022	energy_y
0.0116 ± 0.0015	instrumentalness_y
0.0088 ± 0.0018	liveness_y
0.0058 ± 0.0025	speechiness_y
0.0051 ± 0.0010	tempo_y
0.0026 ± 0.0012	readability
0.0012 ± 0.0010	compound
0.0005 ± 0.0010	key_y
0.0001 ± 0.0001	mode_y

Figure 4. PFI for XGBoost

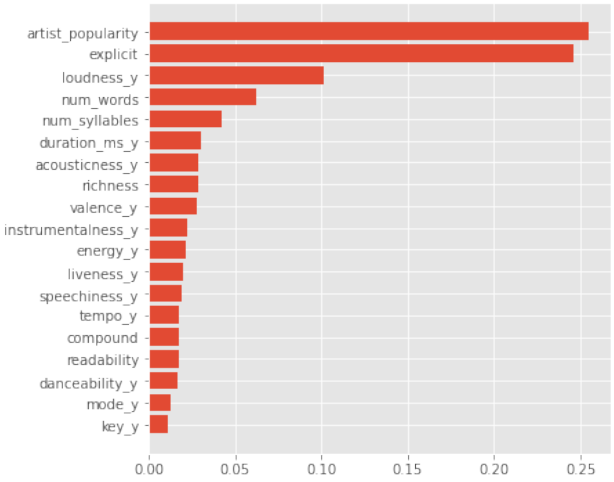

Figure 5. XGBoost Feature Importance

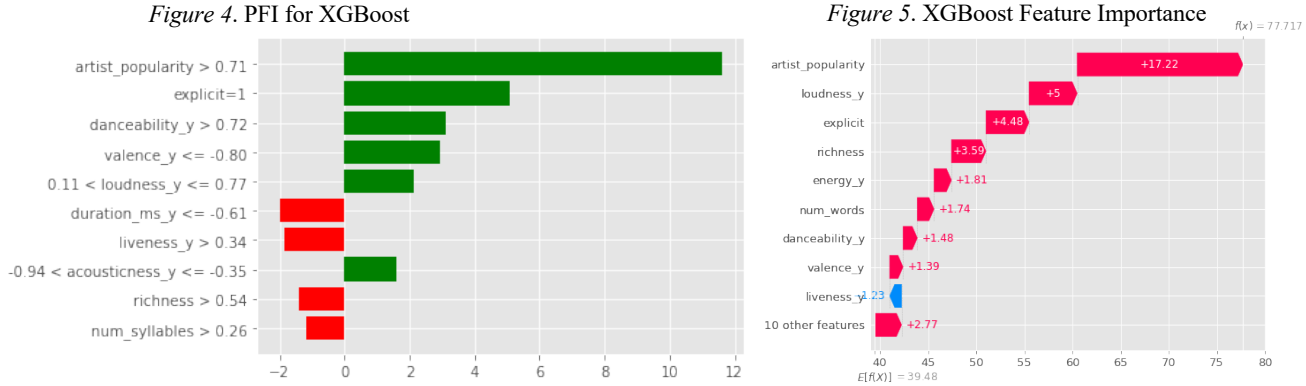

Figure 6. LIME and SHAP of XGBoost for the song with predicted highest popularity score

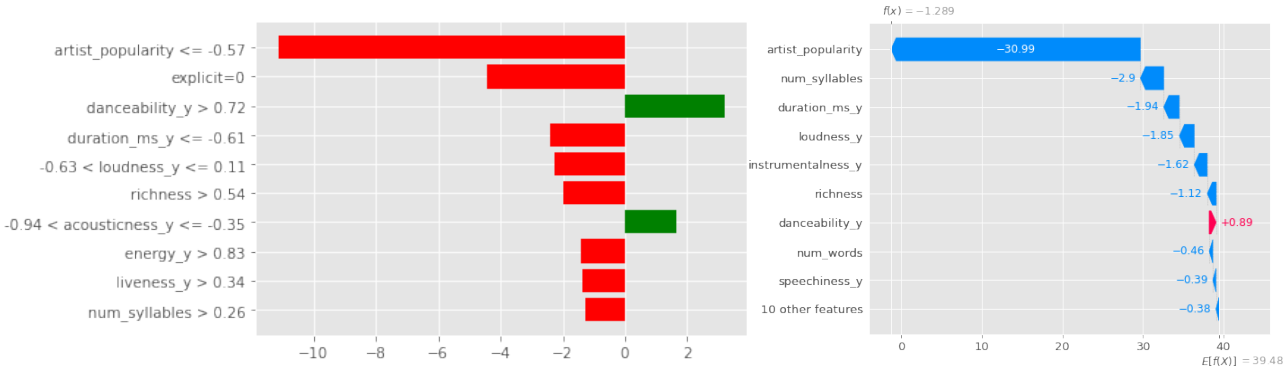

Figure 7. LIME and SHAP of XGBoost for the song with predicted lowest popularity score

Figure 8. Wordcloud plot for popular songs

Figure 9. Wordcloud plot for unpopular songs

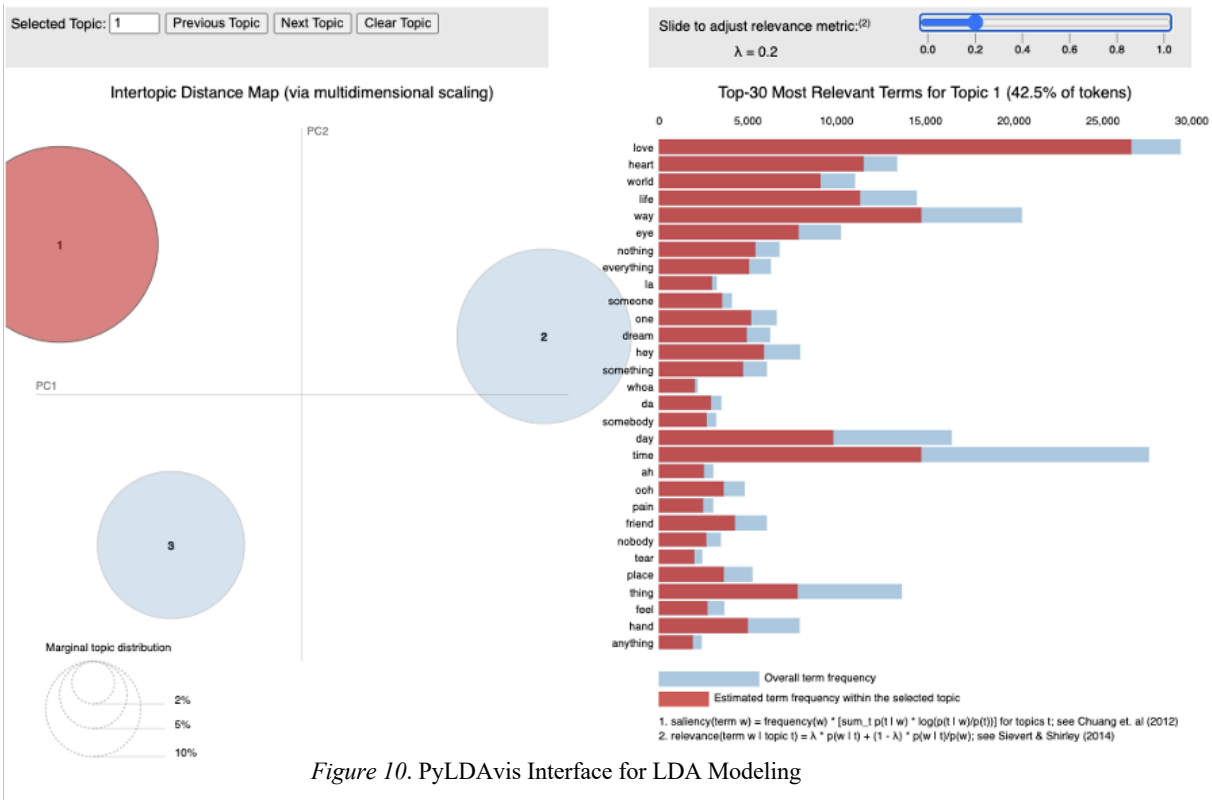

Figure 10. PyLDAvis Interface for LDA Modeling

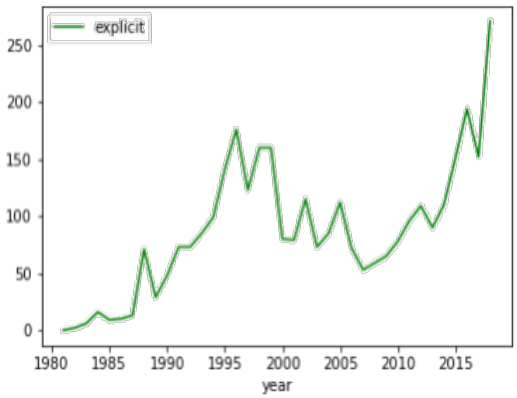

Figure 11. Number of Explicit songs over years

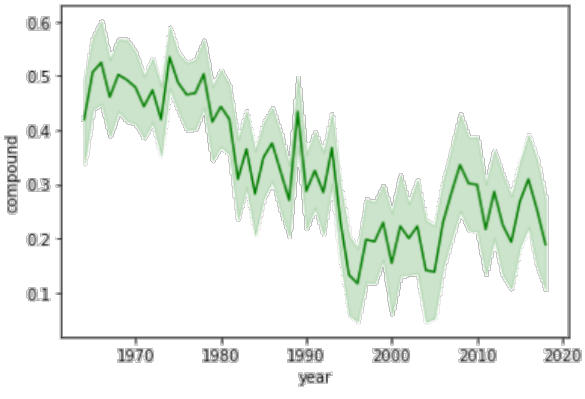

Figure 12. Trend of compound score over years

Table 1 List of Variables

Variable name	*Variable type*	*Description*
Id	String	ID of track generated by Spotify
Name	String	Name of the song
Album	String	Album name that the song belongs to
Artist	String	Artist name of the song
Artist_followers	Integer	Artists' followers on Spotify
Release_date	String	Date of release, mostly in yyyy-mm-dd format, however precision of date may vary
Explicit	Integer	0 = No explicit content, 1 = Explicit content
Mode	Float	0 = Minor, 1 = Major
Popularity	Integer	Measuring the number of streams during a recent short timeframe, ranging from 0 to 100
Duration_ms	Float	Song duration in ms, ranging from 200k to 300k
Key	Float	All keys on octave encoded as values ranging from 0 to 11, starting on C as 0, C# as 1 and so on
Acousticness	Float	Measuring how likely the algorithm thinks it is that this song was recorded acoustically (i.e., without electronic instruments or effects), ranging from 0 to 1
Danceability	Float	Measuring how suitable a song is to dance to, ranging from 0 to 1
Energy	Float	Measuring a track's "intensity", ranging from 0 to 1
Instrumentalness	Float	Estimating the likelihood that this is an instrumental track, ranging from 0 to 1
Liveness	Float	Measuring how likely the algorithm thinks it is that this song was recorded live, ranging from 0 to 1
Speechiness	Float	Measuring the amount of spoken-word (as opposed to singing) in the track, ranging from 0 to 1
Valence	Float	Measuring "positiveness" of the track (i.e., is listening to it likely to make you happy), ranging from 0 to 1
Loudness	Float	Measuring loudness, ranging from -60 to 0
Tempo	Float	Measuring the speed of the song using beats per minute (bpm), ranging from 50 to 150
Popularity_dummy	Integer	Binary, measuring whether the song is popular, (i.e., whether a song was on the Billboard 200)
Lyrics	String	Lyrics of the song, none if it is pure music