Music Moirai

The Cutting Edge “Hit or Flop” Song Predictor ML Model

Matthew Yeon, Neha Gundavarapu, Jenny Belson
ISOM 499R - 1102
December 2, 2021

Introduction / Executive Summary: Record labels receive hundreds of demo submissions daily from random artists across America, each of whom is trying to get discovered as the next "Justin Bieber" or "Beyoncé." The listening process for these demos is very tedious, and many demos are never heard. Further, the demo listeners may be very subjective, potentially causing a label to miss out on a great artist. We realize the potential monetary and recognition value of all-time hits for record labels and that artists' long-term investment is their top priority. Therefore, we propose a popularity predictor and song similarity model for labels to streamline their process and efficiently select the best songs and artists, weeding out demos that don't even deserve a listen. We hypothesized that if we successfully develop such a tool, we could help Eagle Records sign with attractive artists and thus ensure a steady revenue stream. Below are our models' components and how we built them.

Our tool, "Music Moirai," prompts the user to input an artist's song pulled from the Spotify Database API. The model runs a 'popularity test' based on 12 audio features -– danceability, acousticness, energy, instrumentalness, liveness speechiness, tempo, valence, and duration. By evaluating these values, the program determines whether a song would meet the criteria for a "hit," defined by at least one feature on the weekly Billboard Hot-100 tracks list, in a given decade between the 1960s and 2010s. If a song is labeled a "hit," the model tells Eagle Records that the artist's music is noteworthy. This result, plus our Song Similarity Generator, would ultimately inform the label if an artist is worth investing in, what the audience the artist would attract, and how to market the music.

Data Wrangling: To develop our popularity predictor model, we used "The Spotify Hit Predictor Dataset (1960-2019)" from Kaggle to analyze six datasets of songs broken down by decade. We chose this dataset to help Eagle Records identify musical feature trends across the decades to support recruitment of upcoming artists and inform marketing strategies based on similar songs for current artists. Our code involved uploading each decade dataset separately into data frames then merging them into a single, combined data frame of 41,106 songs. We began by browsing the

summary statistics of the numerical attributes and graphing distribution plots, which allowed us to understand the average and range of values of audio attributes these songs had. To ensure the development of a proper model, we confirmed an evenly split dataset of 20,553 “hits” (target = 1) and 20,553 “flops” (target = 0). We then ran a logistic regression model on this dataset to explore the statistically significant features since our outcome hit/flop is a binary dependent variable.

Model Building: Following the data exploration stage, we built a predictive logistic regression model for each decade dataset based on these coefficient findings. Preliminary steps included dropping non-numeric features, subsetting the data, partitioning into train/validation/test sets between training and testing data, and standardizing the data. After running our models, we evaluated each's scores of accuracy, precision, recall, F-1, and confusion matrices. Accuracy scores ranged from 65.7% (1970) to 83.2% (2000) on the test data. Given the multitude of external factors affecting song popularity, these results demonstrate a powerful model for the industry.

After determining an accurate and precise model, we imported Spotify API data using a python module called 'spotipy.' This easy-to-use library provides complete access to all music data provided by the Spotify platform and proved a simpler alternative to using urllib and requests modules on the original Spotify API to call on and index its large amounts of JSON data. The songs we pull from Spotify take the place of software we hope to build in the future that would theoretically analyze audio features regardless of whether it's on Spotify. Therefore, we assume for now that the songs we insert or "search for" in our program are demos from new, emerging artists to be tested for their predicted popularity. We converted the song into a unique ID code to generate a list of the song's features that our model utilizes to predict if a song is a hit or not. The user can choose the decade in which a song would've been a hit or not, and our logistic regression-based program spits out the result. After this, the user can choose to identify any number of songs similar to the inputted "demo." This functionality was based on a calculation of k nearest neighbors, defined as the k songs from the Hit Predictor data whose aggregate song features had the smallest Euclidean

distance to the user's initial song input.[1] This feature would help encourage a label to re-release an album from the 50s that could be popular now.

Challenges & Limitations: One of the most significant challenges we encountered was how to apply our built logistic regression model to predict an inputted song. This issue was resolved by creating a function that trained, fit, and predicted with our LR model in one place. Another hurdle we overcame was embedding error handling mechanisms, especially for user inputs, which we resolved with if-else blocks. One limitation of our model is the lack of account for external factors on music popularity, such as initial popularity of an artist, lyrical content, marketing, and chance. Since musical trends change over time, our prediction model may overlook future changes in music preferences. Another limitation is our use of Spotify's built-in feature-analyzer, meaning we can only input songs available on Spotify. The model seems to be very skewed towards favoring louder, faster, "danceable" tunes. Lastly, our model only can work with one song at a time to predict a "hit or not," so we'd like to look into adding functionality to input entire albums in the future.

Insights: Apart from these limitations, our team derived several helpful insights from our data alone and models. A histogram explored the distribution of four song metrics in our combined dataset. We also graphed trends in popularity of music over time to answer questions such as if acoustic songs were more popular back in the day or if average tempo has changed over time among hits or non-hits (Figure 3) These graphs tell us a lot about emerging popular genres. High acousticness and instrumentalness are less indicative of a hit song, and high energy and a "danceable" song render themselves to a more popular song (Figure 2). Therefore, it's clear that fast-paced and synthetically-sounding or computer-generated-like songs, such as those present in dance-pop, EDM, and hip-hop genres, will be taking the public's attention by storm in current and future times. This is advantageous knowledge for labels like Eagle Records.

[1] https://machinelearningmastery.com/tutorial-to-implement-k-nearest-neighbors-in-python-from-scratch/

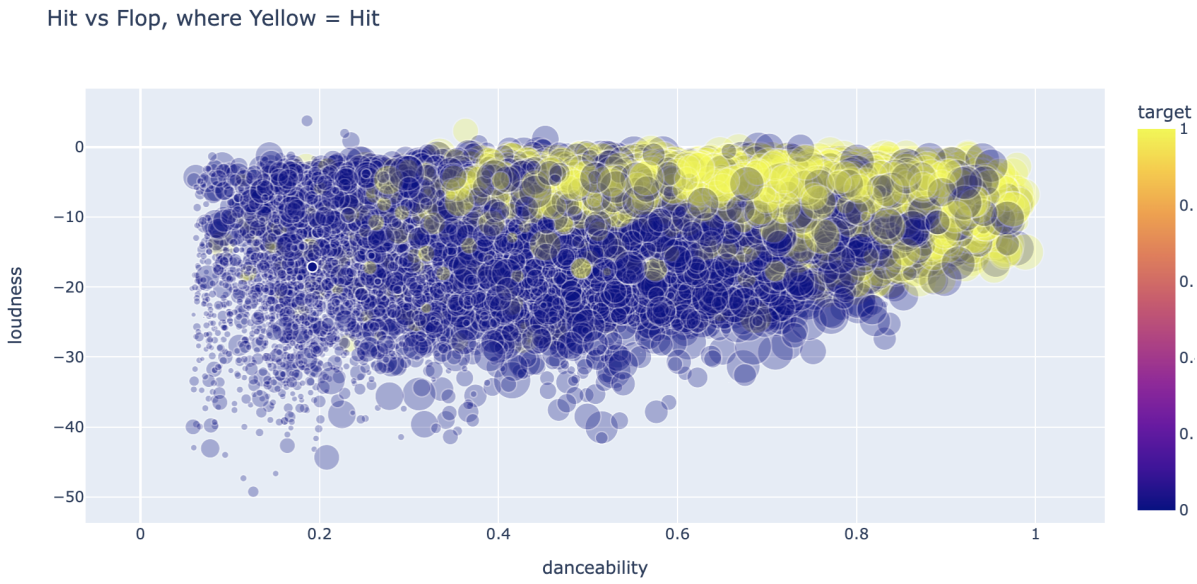
According to renowned econometrician Daniel McFadden, our first exploratory logistic regression model had a pseudo-r-squared of 0.24, which indicates an excellent fit. It confirmed that danceability had the most significant correlation with popularity, as demonstrated by its high relative odds ratio coefficient of 26.02 (the next largest was 1.59). Our next model was a predictive logistic regression model based on the Hit Predictor dataset, which resulted in 60-80%-accuracy in predicting whether a Spotify song would be a hit or not in a specified decade. Following the results of the prediction model, we offer the user a graph informing where the demo falls on the danceability-loudness scale, which are the two most prominent variables that affect the chances of a song being a hit (as per our odds ratio analysis) (Figure 1 & 2).

Discussion: In terms of SWOT analysis, our highly accurate hit prediction model is a strength. Although our current model can only make predictions for a single song at a time and doesn't account for non-numeric song popularity attributes, we see an opportunity to improve our model using music information retrieval systems such as Librosa to acquire a broader spectrum of song features and better predict hit songs. Potential threats for Eagle Record may include Tiktok or even Spotify, all of whom already use advanced algorithms to predict hit songs.

For Eagle Records, a pop music label wanting to sign on artists who will be hits, this model is critical in predicting their success. For most artists, this means we want to predict success in the "2010s" and beyond. However, our decades feature is further helpful for marketing to age groups and re-releasing old albums. For age-group targeting: if a song would've been successful in the 1990s, then it can easily be marketable to an age group of 45-60-year-olds, who would've been teenagers in the 1990s, and thus for whom the song would be nostalgic. For re-releasing old albums, the model inputs old songs to predict a "hit or flop" in this recent decade. Ultimately, our model proves to be multi-purpose in assisting Eagle Records with artist acquisition, (re)release plans, and redefining its future with more informed decisions based on actual facts of how songs performed in the market over the past 80 years. "Music Moirai" is a product that labels cannot find elsewhere.

Appendix

Figure 1: Model-Driven Visualization



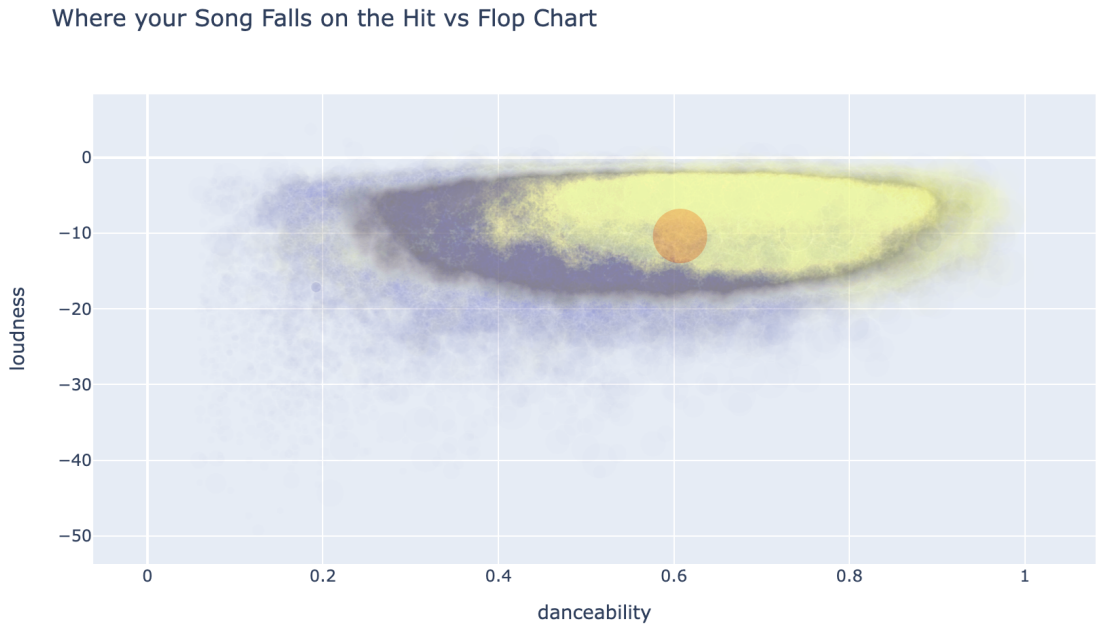
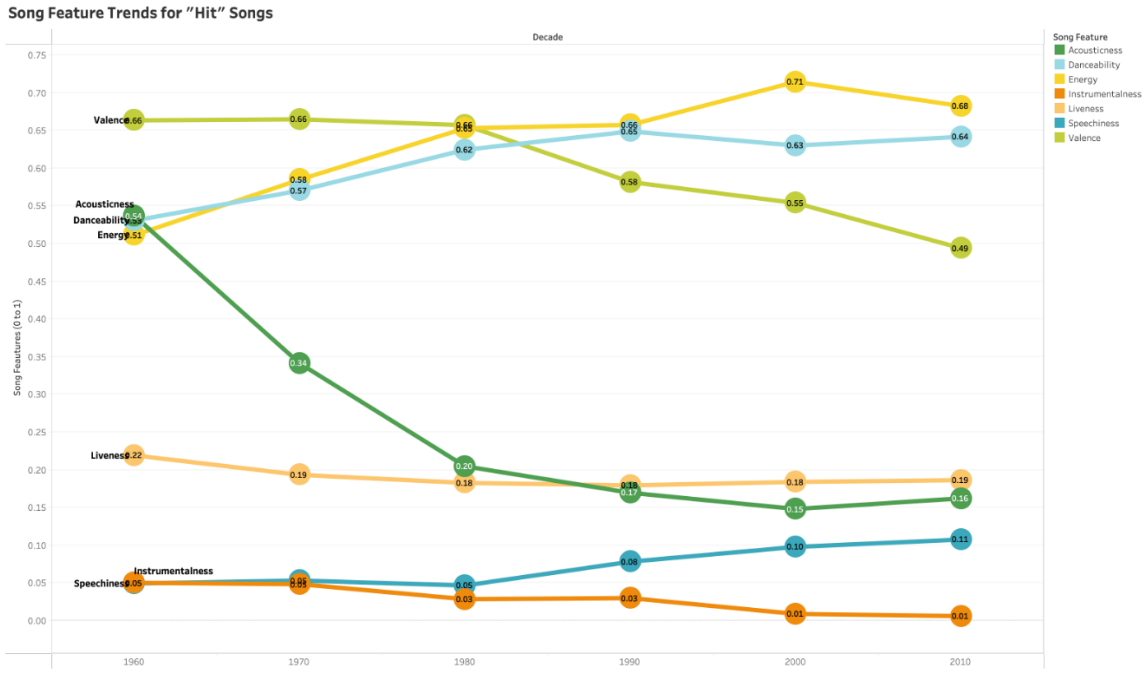
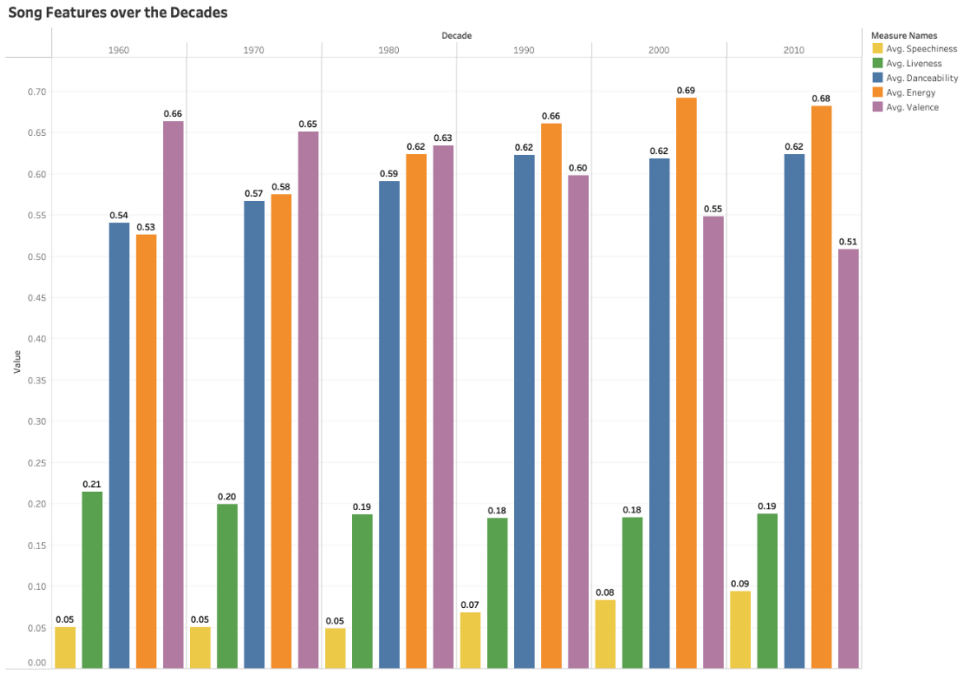


Figure 2: Model-free Visualizations

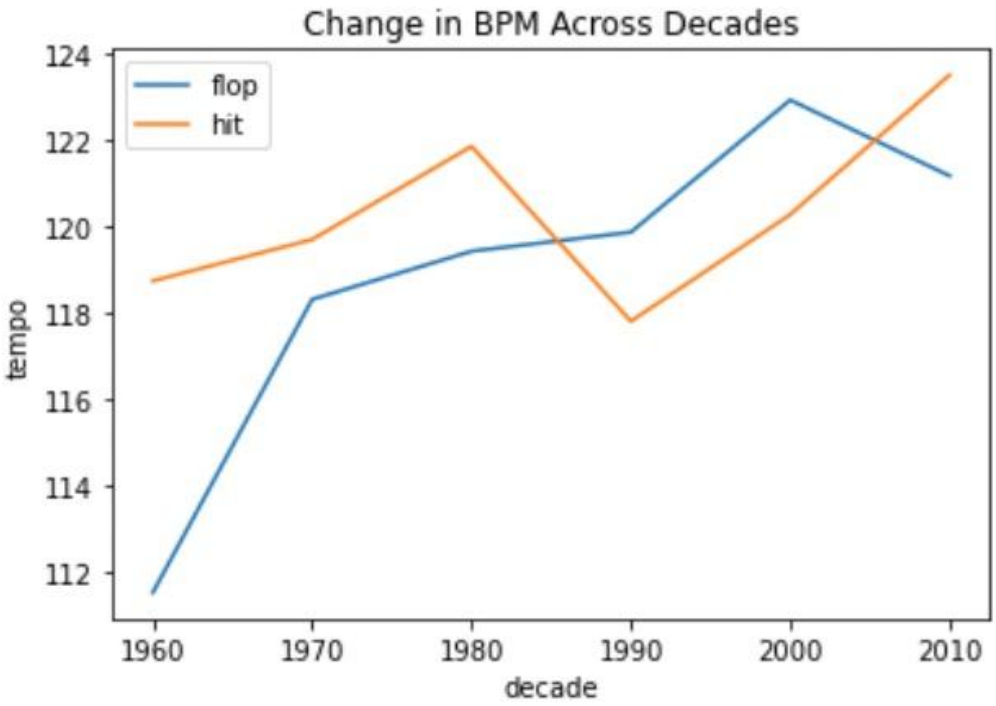


The trends of average of Song Feaure values and average of Song Feaure values for Decade Year. Color shows details about Song Feature. For pane Average of Song Feaure values: The marks are labeled by Song Feature. For pane Average of Song Feaure values (2): The marks are labeled by average of Song Feaure values. The data is filtered on Hit or not?, which keeps True.



Avg. Speechiness, Avg. Liveness, Avg. Danceability, Avg. Energy and Avg. Valence for each Decade Year. Color shows details about Avg. Speechiness, Avg. Liveness, Avg. Danceability, Avg. Energy and Avg. Valence. The marks are labeled by Avg. Speechiness, Avg. Liveness, Avg. Danceability, Avg. Energy and Avg. Valence. The data is filtered on Hit or not?, which keeps True.

Figure 3


Change in BPM Across Decades
flop
hit
tempo
124
122
120
118
116
114
112
1960
1970
1980
1990
2000
2010
decade

Citations

- Documentation for code:
    - https://spotipy.readthedocs.io/en/2.19.0/#examples

- Developer dashboard: spotify
    - https://developer.spotify.com/dashboard/applications/12a274d91fb54d6f95f9ab4589be1d48

- Examples of uses for spotify API:
    - https://github.com/plamere/spotipy/tree/master/examples
    - https://machinelearningmastery.com/tutorial-to-implement-k-nearest-neighbors-in-python-from-scratch/