Spotify feat. Logistic Regression - Popularity, Nothing Else Matters

Sebastian Hoffmann[*]
University of Tübingen
Matriculation number 5954377

Yannick Streicher[†]
University of Tübingen
Matriculation number 5331817

Abstract

What is the musical taste of the world? With the recent rise and global pervasiveness of music streaming services, such as Spotify, or Apple Music, answering this question has potentially become tractable. For this, we analyze a subset of 1.2 million songs scraped from Spotify that we further augment with popularity and genre information. While we find that the musical features extracted by Spotify are discriminative of the genre of a song, they are poor predictors of the popularity if used in a linear model, such as linear or logistic regression. Our results indicate that either more complex models are necessary or that the task itself is to some degree ill-posed, for instance due to a low signal-to-noise ratio.

1 Introduction

With the recent advent of musical streaming services, such as Spotify, or Apple Music, unprecedented amounts of music data became readily available to the public. This data can often be accessed free-of-charge using an API provided by the provider of the service itself. Due to their large userbase, popularity measures derived from these platforms provide a robust measure, at least for their target demographics.

In this work we want to explore if the musical features extracted by Spotify for individual songs, such as tempo or key, can be used to predict its popularity. We are particular interested in if this can be achieved using a simple linear model such as linear or logistic regression. This would allow direct interpretability of what musical properties a popular song possesses. Furthermore, we are interested in how todays musical landscape is shaped. To this end, we analyze the distribution of genres of songs available on Spotify and investigate if there are some genres which are particular popular or unpopular.

2 Dataset

While Spotify provides data about individual songs or artists via its API, it does, however, not provide a catalog of available songs on its platform. Thus, we use a dataset[3] of 1.2 million songs made available on Kaggle, a subset of the 70 million songs accessible on Spotify [2]. This dataset was created by first downloading the MusicBrainz catalog, an open-collaboration database of music releases, and then querying the Spotify API.

For each song listed, the dataset contains basic meta informations such as artist and song name, as well as a set of 14 song features that are provided by Spotify. These features include, among others,

[*]correspondence to sebastian.hoffmann@student.uni-tuebingen.de
[†]correspondence to yannick.streicher@student.uni-tuebingen.de
[3]https://www.kaggle.com/rodolfofigueroa/spotify-12m-songs

the estimated tempo, key, energy, danceability, or duration of the song. For a full list, refer to the Spotify API documentation. Crucially, the popularity or genre of a song is not included in this dataset.

2.1 Augmentation and Filtering

In a first step we retrieve additional information of all $85\,113$ artists appearing in the dataset from the Spotify API. This information includes the popularity of that artist, aggregated from individual songs, the number of followers, and optionally a list of genres the artist is associated with. Additionally, we query the Spotify API for every song in the dataset to obtain its individual popularity. While Spotify does not provide genre information at song level, we use the genre information associated with the first artist to further augment the dataset. The exact procedure is described in Section 2.2. We further apply a filtering scheme based on PCA to only include professional or semi-professional musicians. Further details are given in the supplementary material on github[4]. In total, 62.8% of all artists are removed from the dataset. This reduces the number of songs left in the dataset to $755\,472$ (62.75%).

2.2 Finding supergenres by clustering

The genres associated with an artist are often finely granulated. Out of 4713 genres in total, only 496 have more than 50 artists associated with them. To reduce the number of overall genres to a handful of overarching *supergenres*, such as Rock, Jazz, or Hip-Hop, we employ agglomerative clustering [7] (we use [4]). Agglomerative clustering lends itself naturally for this as it exploits the inherent hierachical relationship between genres.

Inspired by constrastive methods [3, 1], we exploit the fact that related genres are more likely to be associated with the same artist to construct a distance measure. In a first step, a similarity measure $s_{i,j} = (2n_{ij})/(N_i + N_j)$ between the i-th and j-th genre is defined, where n_{ij} denotes the number of times they appear together and N_i, N_j the total counts of the i-th and j-th genre respectively. The distance can then be defined as the reciprocal of that similarity measure. We then use this distance measure for the clustering procedure.

By running the clustering algorithm on an aggressively filtered subset of genres, we find 32 supergenres, among them Jazz, Classical, Metal, Rock, Pop, or EDM, but also Soundtrack, or Show Tunes[5]. These cover 60.5% of all songs in the filtered dataset. To further improve coverage of songs, we rerun the clustering with less aggressive filtering to include more fine-grained genres. We then manually append these additional subgenres to the appropriate supergenre if applicable. This improves coverage to 71.3%.

3 Experiments

3.1 Genre analysis

The most prelevant genre on Spotify is Classical (20.6%), followed by Rock (7.9%), Rap / Hip-Hop (7.6%), Metal (6.3%), Folk (6.0%), and Jazz (4.0%). For a complete listing, we refer to the supplementary material available on github[5].

The box plot presented in Fig. 1 shows the distribution of popularity for each supergenre. We find that despite being the most prominent genre, Clasical and Contempory Classical are by far the most unpopular with the majority of the probability mass below 10%. Notably, Pop has by far the heaviest tails and the highest median and 75-th percentile of all genres despite only making up 2.9% of all songs. Following after Pop, we find that Country, EDM, Latin, Rock, Indie Pop, Hip-Hop, and Metalcore are almost equally popular.

Lastly, as a preliminary experiment for the logistic regression, we investigate the discriminative power of the musical features provided by Spotify. For this, we embed the features of songs of the six most prominent genres into a two-dimensional vector space using t-SNE. Results are presented in Fig. 3 in the appendix. Clearly visible are distinct cluster of songs corresponding to genres such as Hip-Hop, Metal, Classical or Jazz. Additionally, genres that are arguably to some extent similar, such as Rock and Metal, or Jazz and Classical, are clustered close together.

[4] https://github.com/ystreicher/SpotifyNEM2022

[5] Supergenres have been manually annotated

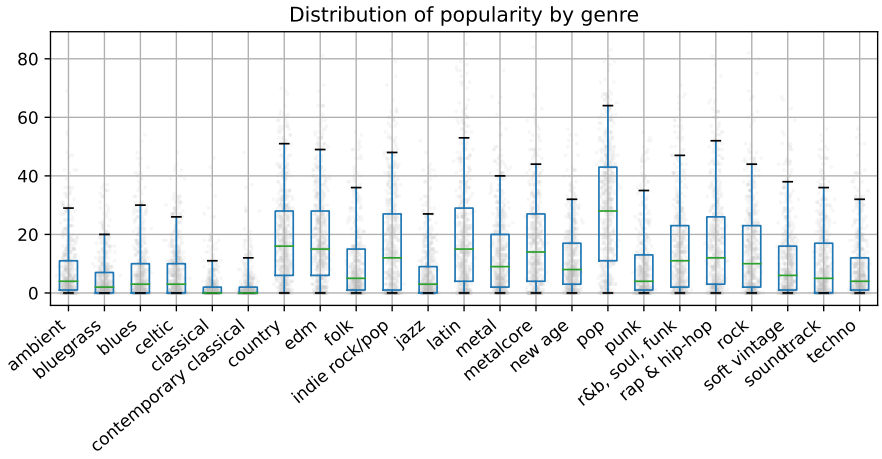

Figure 1: Box plot of popularity grouped by genre. Boxes span $Q1$ to $Q3$,while whiskers indicate the 5-th and 95-th percentile respectively. The central line shows the median of the distribution. Among all genres, Pop has the highest median and 95-th percentile.

3.2 Logistic Regression

Before applying logistic regression, we test if popularity can be predicted using linear regression from the musical features (c.f. Section 2) as a preliminary experiment. We obtain significant but small coefficients that can only explain about $10\,\%$ of the variance in the data. Loudness appears to be the most important feature according to the model. We now investigate if logistic regression can be used to classify songs as either popular or unpopular.

Binary labels are obtained by thresholding the popularity value of a song. Despite our aggressive filtering, we find that the popularity is roughly exponentially distributed. To deal with such a highly-skewed distribution, we try two thresholding strategies:

(A) Any song with popularity below the 90-th percentile is labelled *unpopular*, otherwise as *popular*. To counteract the class-imbalance, the class weights are adjusted.

(B) The songs are evenly split into two classes at the 50-th percentile.

In all experiments an 80-20 train-test split is used, and ℓ_2-regularizaton is employed in both cases. To find the optimal regularization strength, we performed a grid-search using 5-fold cross validation on the training set.

Overall, model A achieves an accuracy of 0.59, whereas model B achieves an accuracy of 0.62 w.r.t. to their respective labels. A calibration plot is presented in Fig. 2 (left). Both models are poorly calibrated and possess higher mean ℓ_1-loss than random guessing (not shown). Combined with the low R^2-score of the linear model, this might indicate that popularity can only be poorly predicted using a linear relationship. To test if a higher capacity model could potentially improve performance, we further train a simple multilayer perceptron, but report only marginally improved results.

Nevertheless, we show learned coefficients of model B in Fig. 2 (right). In accordance with the linear regression, especially loudness and danceability seem to be indicative of popularity. In contrast, acousticness, instrumentalness, energy, and valence appear to have a negative impact.

4 Conclusion

In this work, we gave an overview of the musical landscape as seen by Spotify. We demonstrated that agglomerative clustering can successfully be applied to aggregate the thousands of microgenres identified by Spotify's algorithm into a handful of supergenres. Furthermore, we have shown that the musical features provided by Spotify's API are highly discriminative in terms of *musical relatedness* by applying t-SNE and showing that songs of the same genre are spatially close in feature space.

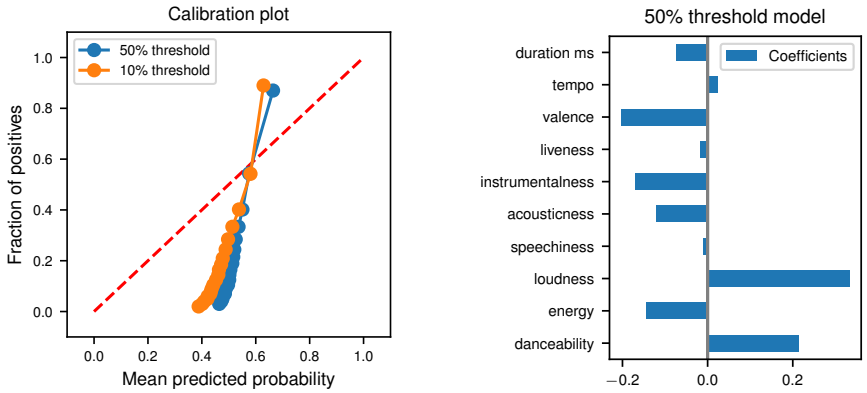

Figure 2: *Left*: Calibration plot. The prediction space is discretized into 20 bins. Using the test set, for each bin, we plot the fraction of true popular songs against the mean predicted probability for those songs. *Right*: The coefficients of the selected model for each audio feature. For the interpretation of individual features consider the API documentation of Spotify.

Despite some genres being less and more popular than others, we find linear models to be only poor predictors of the popularity of an individual song.

Contrary to previous results [6] ($n = 30$), our experiments show that loudness correlates, albeit very weakly, with the popularity of a song. This does not necessarily imply a causation, however. Rather, it could be indicative of the previously recognized *loudness war* [5]. Either, because successful songs are more likely to have undergone a rigorous mixing and mastering process, or because more recently released songs, thus on-average louder songs, are more popular by construction of the metric.

Our results provide some evidence that the musical properties of a song might only have a minor impact on its popularity and commercial success. As previously recognized in the literature [6], other factors not covered in the data, such as marketing efforts, or radio and television airplay might have a much bigger impact instead. Finally, it is not unreasonable to assume that popularity is inherently very noisy, i.e. dominated by chance.

References

[1] T. Chen, S. Kornblith, M. Norouzi, and G. Hinton. A simple framework for contrastive learning of visual representations. In *International conference on machine learning*, pages 1597–1607. PMLR, 2020.

[2] T. Ingham. Spotify now hosts 70 million songs. but it can't keep that up forever, Nov 2020. URL https://www.rollingstone.com/pro/features/spotify-now-hosts-70-million-songs-but-it-cant-keep-that-up-forever-1094234/.

[3] T. Mikolov, K. Chen, G. Corrado, and J. Dean. Efficient estimation of word representations in vector space, 2013.

[4] F. Pedregosa, G. Varoquaux, A. Gramfort, V. Michel, B. Thirion, O. Grisel, M. Blondel, P. Prettenhofer, R. Weiss, V. Dubourg, J. Vanderplas, A. Passos, D. Cournapeau, M. Brucher, M. Perrot, and E. Duchesnay. Scikit-learn: Machine learning in Python. *Journal of Machine Learning Research*, 12:2825–2830, 2011.

[5] E. Vickers. The loudness war: Background, speculation, and recommendations. In *Audio Engineering Society Convention 129*. Audio Engineering Society, 2010.

[6] D. Viney. The obsession with compression. *Master of Music Technology Thesis, London College of Music, Thames Valley University*, 2008.

[7] J. H. Ward Jr. Hierarchical grouping to optimize an objective function. *Journal of the American statistical association*, 58(301):236–244, 1963.

A Appendix

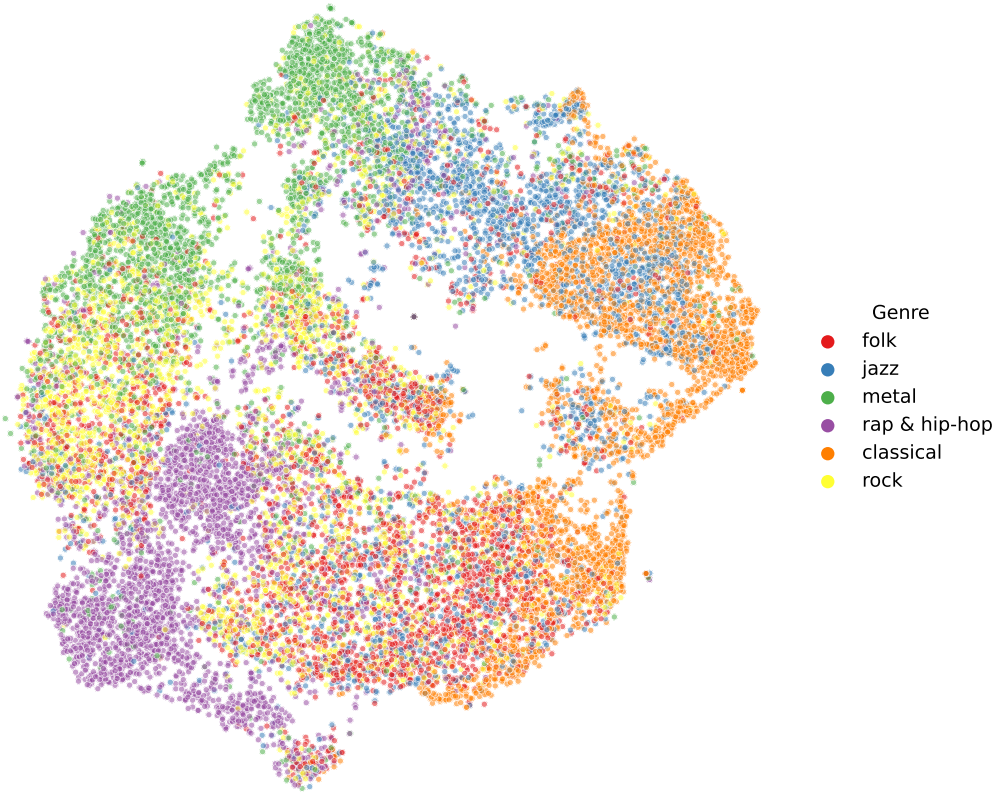

Figure 3: Two-dimensional embedding of song features results in distinct clusters corresponding to musical supergenres.