Regression-based Prediction of Song Popularity

Jessie Wong

21/01/2022

Contents

Executive Summary

Millions of new songs are released every year, but only a small fraction of them become popular. Some songs easily climb to the top of music popularity charts such as Billboard, whereas other songs sit at the bottom of the charts, and some songs do not gain a spot on these charts at all. With endless new artists constantly emerging in the booming music industry, it has become increasing competitive for artists to release the most popular songs. Having an algorithm that predicts how popular a new song will be can be an effective tool for artists writing new music that they hope will become major hits. This report details the process and results of the construction and development of a machine learning algorithm to predict the popularity score of a song given various features such as song duration, genre, title, and tempo. We use a Ridge algorithm to build a simple linear regression model, and we study the cross-validation scores generated by the machine learning model. We also carried out hyperparameter optimization to determine the parameter values that would yield the highest accuracy. The results show that the trained algorithm correctly predicts the popularity score of a song 51% of the time, and we were able to determine that duration, loudness and speechiness of a song have the largest impact on the popularity score. Further development and fine-tuning of the algorithm is needed to improve its prediction accuracy score, but even with an algorithm of high accuracy, a machine learned model is restrained by the scope of information used in the training data. The dataset used in this project was sourced from Spotify and Billboard Charts based on song popularity in the United States, therefore the model may produce varying results if applied to data from different demographics. The model also only predicts a popularity score as perceived by a wide audience, rather than any individual preference. In this report we delve into the detailed analysis process that was undertaken to produce the machine learning model, and we also discuss the limitations of the process and outcomes.

Introduction

According to a report from Music Business Worldwide linked here, approximately 137 million new songs are released every year, but only about 14 records have sold 15 million physical copies or more in global history. Of the millions of new songs, there is an endless selection of music of all different styles. We can categorize song styles based on characteristics such as their genres, tempos, loudness, and acousticness. With so many songs in the world, one can't help but wonder why some types of songs seem to reach everyone's ears, while others are unheard of. This poses an interesting question on what exactly makes a song popular, and whether we can train a model to identify trends in characteristics that are associated to greater popularity of a song, and use those trends to predict the popularity of a new song.

In this project we used a dataset containing information for nearly 30 thousand Spotify tracks, and built an algorithm to predict the popularity of a song based on features such as genre, tempo, loudness, and acousticness. We used a Ridge algorithm to build a linear regression model to predict the popularity score, which ranges from 0 to 100. A popularity score of 0 indicates that the song has no popularity and a popularity score of 100 means the song is extremely popular. By using a linear regression model, we operated under the assumption that there is some linear correlation between between song characteristics and popularity. While this approach is feasible for predicting the popularity of a song, the linearity assumption may not fully capture the trends in the data. Furthermore, the dataset that was used in this project contained only songs available on Spotify and the popularity scores were sourced from Billboard Charts, which is a United States based music chart. The performance of a machine learning model is limited by the data that was used to train it, therefore the model trained in this project may not be applicable to songs from different geographic locations.

More traditional approaches have historically been used to gauge song popularity, such as through widespread distribution of the music after release and collecting listener feedback. This feedback is commonly summarized through music charts such as Billboard. An advantage to having a ML model predict the popularity of a song is that the feedback is instant and the technology can be used even before a song is released so that the artist can make adjustments to the song to increase predicted popularity.

Methods

Data Collection

The dataset used in this project was sourced from Tidy Tuesday's github repo here, and the specific dataset that was used is linked here. The data originally comes from Data.World, Billboard.com and Spotify. The popularity scores in the dataset were sourced particularly from Billboard Charts, which is the music industry standard record chart in the United States for songs. Chart rankings, and thus the popularity scores, are based on physical and digital sales, radio play, and online streaming in the United States. This limits the scope of the model to only be able to predict the popularity of new songs based on the learned preferences in music of United States population.

The dataset contains nearly 30 thousand rows, and 21 columns. Each row from the dataset represents one song, and each column represents a feature. The target column is the track popularity specifying the song's popularity score on a scale from 0 to 100. The features in the dataset are song ID, performer, song, track genre, track id, track preview url, track duration ,track explicit, track album, danceability, energy, key, loudness, mode, speechiness, acousticness, instrumentalness, liveness, valence, tempo, and time_signature. A description of each column in the dataset and their data type is listed in the table below.

Table 1: Feature Descriptions

Feature	**Data.Type**	**Description**
song_id	character	Unique ID for the song
performer	character	Performer
song	character	Song title
spotify_genre	character	Song genre(s)
spotify_track_id	character	Unique ID of song on Spotify platform
spotify_track_preview_url	character	URL to song on Spotify
spotify_track_duration_ms	numeric	Song duration in ms
spotify_track_explicit	logical	True if song is explicit, False if not
spotify_track_album	character	Album name
danceability	numeric	Describes how suitable a track is for dancing based on a combination of musical elements including tempo, rhythm stability, beat strength, and overall regularity. A value of 0.0 is least danceable and 1.0 is most danceable
energy	numeric	Is a measure from 0.0 to 1.0 and represents a perceptual measure of intensity and activity. Typically, energetic tracks feel fast, loud, and noisy. Perceptual features contributing to this attribute include dynamic range, perceived loudness, timbre, onset rate, and general entropy.
key	numeric	The estimated overall key of the track. Integers map to pitches using standard Pitch Class notation
loudness	numeric	The overall loudness of a track in decibels (dB). Loudness values are averaged across the entire track. Values typical range between -60 and 0 db.
mode	numeric	Indicates the modality (major or minor) of a track, the type of scale from which its melodic content is derived. Major is represented by 1 and minor is 0.
speechiness	numeric	Detects the presence of spoken words in a track. The more exclusively speech-like the recording (e.g. talk show, audio book, poetry), the closer to 1.0 the attribute value.
acousticness	numeric	A confidence measure from 0.0 to 1.0 of whether the track is acoustic. 1.0 represents high confidence the track is acoustic.
instrumentalness	numeric	Predicts whether a track contains no vocals. The closer the instrumentalness value is to 1.0, the greater likelihood the track contains no vocal content. Values above 0.5 are intended to represent instrumental tracks, but confidence is higher as the value approaches 1.0.
liveness	numeric	Detects the presence of an audience in the recording. Higher liveness values represent an increased probability that the track was performed live. A value above 0.8 provides strong likelihood that the track is live.
valence	numeric	A measure from 0.0 to 1.0 describing the musical positiveness conveyed by a track. Tracks with high valence sound more positive (e.g. happy, cheerful, euphoric), while tracks with low valence sound more negative (e.g. sad, depressed, angry).
tempo	numeric	The overall estimated tempo of a track in beats per minute (BPM). In musical terminology, tempo is the speed or pace of a given piece and derives directly from the average beat duration.
time_signature	numeric	Time Signature
spotify_track_popularity	numeric	Popularity

An initial exploratory data analysis was performed to investigate the distribution of the target column. The plot of the distribution is shown below. We can observe from the plot that the distribution is heavily left skewed, meaning there are many more songs with low popularity scores than high popularity scores. This introduces further bias in the model, since there will be more examples of lower popularity songs for the model to learn from. This potentially influences the model to in turn predict lower popularity scores on new songs, since it has learned many characteristics of low popularity songs, but fewer characteristics of high popularity songs. However, this distribution also accurately represents what we know about song popularity charts. Very few songs make it to the top, while most songs have low popularity and never rank on the charts.

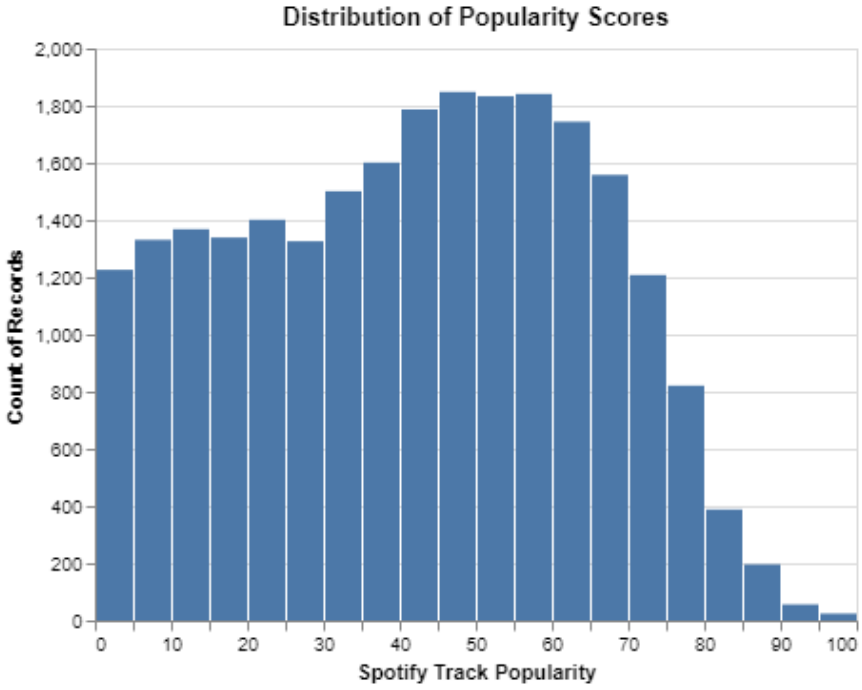

Figure 1: Distribution of Popularity Values is Left Skewed

Data Preprocessing

To clean up the data, we removed the columns 'song_id', 'spotify_track_id', 'spotify_track_album', and "spotify_track_preview_url". Song ID, track ID and preview URL do contain any relevant information about the song, and they would not be useful to the prediction algorithm. Track album was also dropped because there were too many unique values in the column, and we would not be able to extract any meaningful value. We then also dropped rows that had missing data. After the data cleaning, we resulted in a dataset containing around 24000 examples and 17 features. The data was then split into training and testing data, with 80% of the data in the training set and 20% in the testing set. Because we have a large dataset on hand, we are not at risk of overfitting the validation set.

The feature columns were categorized as shown in the table below, in preparation of building a column transformer to preprocess the data before analysis. The data was then preprocessed using a column transformer, specifically by using a Standard Scaler on the numeric features, One Hot Encoder on the binary and categorical features, and a Count Vectorizer on the text features. This was done for both the training and testing datasets.

Table 2: Feature Column Types

Type	**Features**
Numeric	spotify_track_duration_ms, danceability, energy, key, loudness, mode, speechiness, acousticness, instrumentalness, liveness, valence, tempo,time_signature
Categorical	performer
Binary	spotify_track_explicit
Text	spotify_genre, song
Drop	song_id, spotify_track_id, spotify_track_album, spotify_track_preview_url
Target	spotify_track_popularity

To determine which key features would potentially have the largest impact on the popularity score, we made a paired distribution and correlation plot of the numeric features, which can be found in Appendix A. From the plot we can observe that duration, speechiness and loudness have the strongest correlations with popularity. In addition to using the plot as a quick interpretation of which numeric feature had the highest correlation to popularity score, we were also able to use the plot to determine if there was any collinearity between numeric features. It is usually very important to examine if and how features are correlated and to investigate their pairwise distributions. In the plot, we can see that the bottom left half the plots show the correlation plots and the upper right half show the correlation values. The diagonal line of plots from top left to bottom right show the distributions of each feature.

Data Analysis

We adopted a linear regression model for this problem, specifically using a Ridge algorithm. Our choice of Ridge stems from the fact it is regularized and takes care of any multi-collinearity issues. It uses L2 regularization, meaning that the weights will be equally distributed among all collinear features because the solution is unique. The model was then fitted and trained on the training dataset, and a 10-fold cross validation was carried out. The train and validation scores for each fold are reported in the table below. Model performance was evaluated using the training and test scores. As can be seen from the cross-validation scores below, the model did not perform exceptionally well. The train scores averaged at 0.79, while the test scores averaged around 0.47. The scores were quite low and there was also evidence of overfitting in this model since the train scores are consistently larger than the validation scores by a moderate amount.

Table 3: Train and validation scores from cross-validation

Fold	**Validation score**	**Train score**
1	0.48	0.79
2	0.47	0.79
3	0.50	0.79
4	0.48	0.79
5	0.45	0.79
6	0.48	0.79
7	0.47	0.79
8	0.48	0.79
9	0.46	0.79
10	0.50	0.79

We improved accuracy scores through hyperparameter optimization. We opted to evaluate the effects of different values of the alpha parameter in Ridge, since the alpha value controls the fundamental tradeoff in machine learning, and can impact whether a model is underfitting or overfitting. We also optimized the values for max features and binary in the two Count Vectorizers, because the implementation of the Count Vectorizers have a large impact on the number of features used, which can impact model performance. We chose to use a randomized search method for optimization over grid search because we were testing out many parameters, all with many different possible values. A grid search would be too computationally demanding in terms of memory and time compared to a randomized search. The following table shows the results of the RandomizedSearchCV function for determining the best hyperparameters for the Ridge model and Count Vectorizer preprocessors. CV-1 represents the Count Vectorizer used on the genre feature, and CV-2 represents the Count Vectorizer used on the song titles feature.

Table 4: Best hyperparameters from RandomizedSearchCV

Rank	**Mean test score**	**alpha(Ridge)**	**Max features(CV-1)**	**Binary(CV-1)**	**Max features(CV-2)**	**Binary(CV-2)**
1	0.50	1.0	1000	TRUE	1000	FALSE
2	0.49	1.0	1000	FALSE	1000	FALSE
3	0.49	10.0	1000	TRUE	1000	FALSE
4	0.46	0.1	1000	TRUE	1000	FALSE
5	0.45	0.1	1000	FALSE	1000	FALSE
6	0.45	0.1	1000	FALSE	1000	TRUE
7	0.44	100.0	1000	TRUE	1000	FALSE
8	0.44	100.0	1000	FALSE	1000	FALSE
9	0.44	0.0	1000	TRUE	1000	TRUE
10	0.44	0.0	1000	FALSE	1000	FALSE

From the hyperparameter optimization, the RandomizedSearchCV funciont yielded that that the highest mean train score arose from using an alpha value of 1, max features of 1000 for both Count Vectorizers, and having a binary = True argument and binary = False argument for the 2 Count Vectorizers respectively. Once these optimal hyperparameters were determined, the parameters were used in the Ridge model to once again carry out cross-validation on the training data. The new train and validation scores for each fold are reported in the table below.

Table 5: Improved train and validation scores after hyperparameter optimization

Fold	**Validation score**	**Train score**
1	0.49	0.73
2	0.49	0.73
3	0.51	0.73
4	0.50	0.73
5	0.47	0.73
6	0.49	0.73
7	0.49	0.73
8	0.51	0.73
9	0.49	0.73
10	0.52	0.73

From the results we see that the validation scores have improved marginally, now averaging at 0.50, while the train scores have decreased to an average of 0.73. The hyperparameter optimization resulted in less overfitting of the model, and a slight improvement in the model accuracy performance. The code used to perform this analysis can be found on the project repo linked here.

Results

Using the trained model with tuned hyperparameters, the algorithm was then evaluated on the test data, which yielded an accuracy of 51%. The model was used to predict the popularity of the songs in the test dataset, and predicted values were plotted against the actual popularity values. The result can be seen in the plot below. Many of the data points do not lie on the line, which can be expected given the accuracy, and train and test scores that were fairly low. However, there is a clear trend that the data points follow the trend of the line, which shows that the Goodness of Fit below is not unreasonable and demonstrates the viability of the Ridge model.

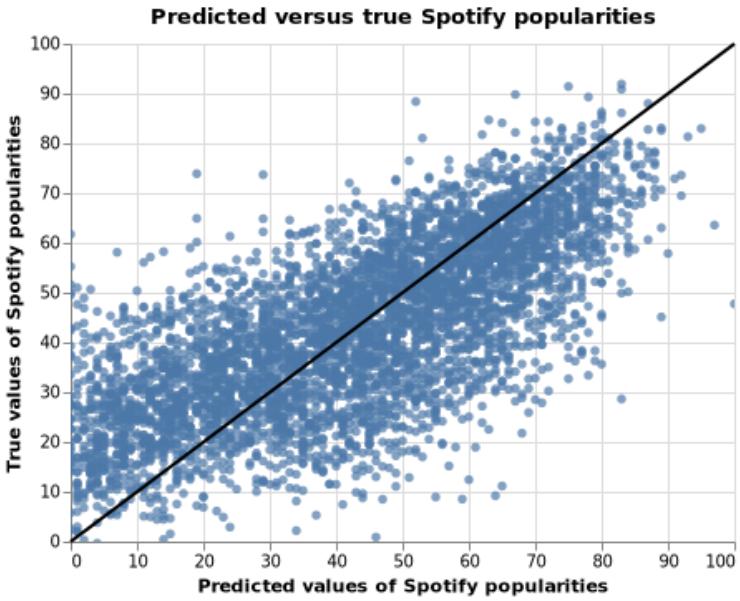

Figure 2: Predicted vs true popularity scores

Conclusion

The trained machine learning model was able to accurately predict the popularity of songs 51% of the time. This leaves much room for improvement before being deployed to the music industry. There are several suggested changes that can be made to the analysis process that could improve the model and obtain higher accuracy on the model predictions. Firstly, the data that was used here was solely based on Spotify and Billboard information based in the United States. To create a more robust model, data could be collected from a wider range of sources from other music platforms and countries around the world. The scope of this project was limited by the data that was used, but expanding the training set could broaden the usage of the machine learning model to users outside of the United States. Another setback of the project is that the trained Ridge model did not perform very well overall, and further model development is likely needed to improve performance. A possible revision for this is to carry out feature engineering to create useful features to yield a better performing model. The preprocessing for text features could also be changed to a encoding more sophisticated than a Count Vectorizer, since it is not a great representation of language. It discards any syntax and compositional meaning in the language, which causes us to lose a lot of information about the text. This could have a large impact on song titles, as they are often metaphorical and contain deeper meanings than individual words. Overall, although the reported model does not perform exceedingly well, it serves as a good baseline for developing a machine learning model to predict song popularity. Further developments leading to a high accuracy model could have large impacts on the music industry in the future.

Appendix

Appendix A: Pairwise Distribution Plot

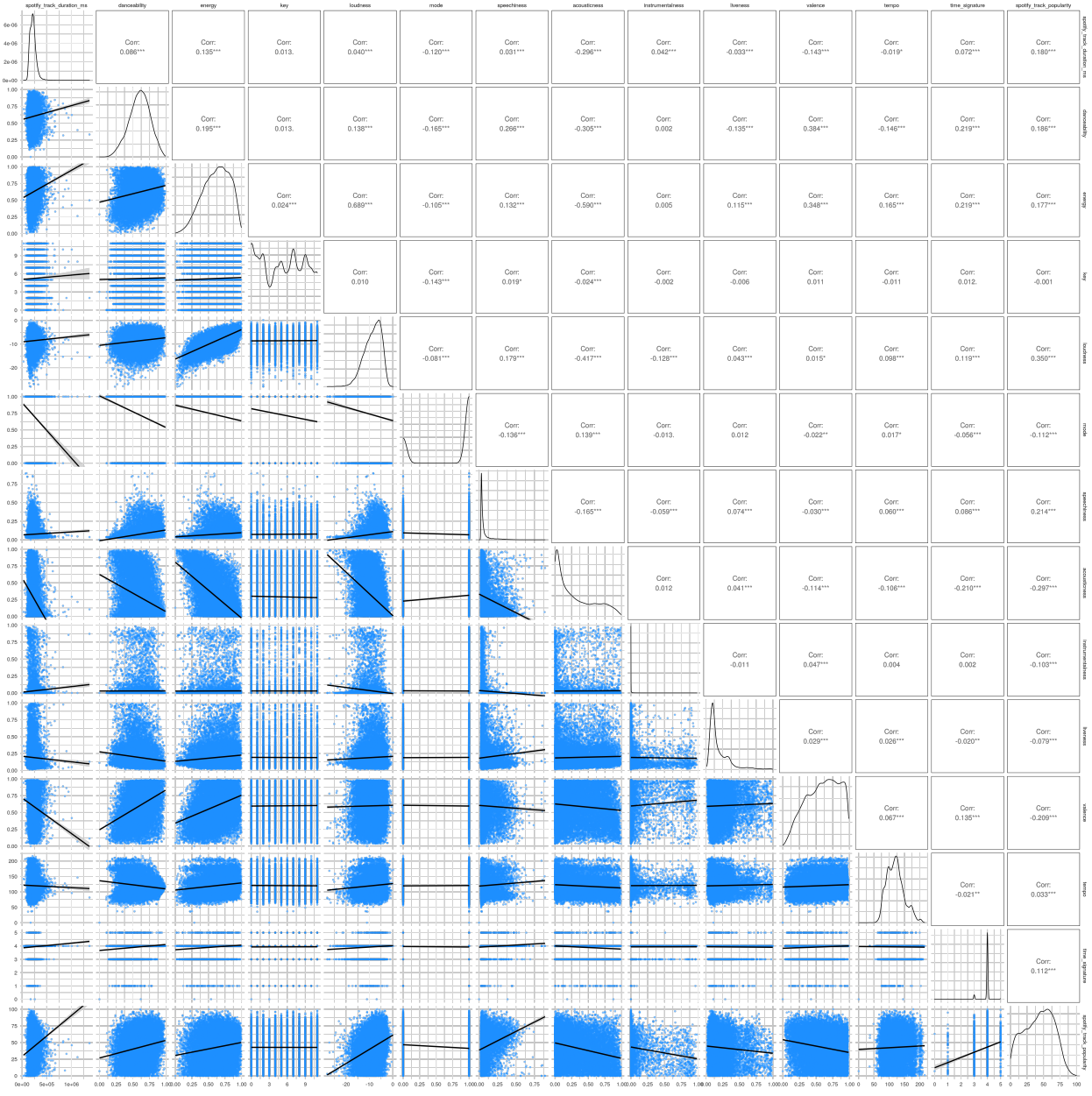

Figure 3: Pairwise distributions and correlations of numeric features