Genius Lyrics Genre Prediction

Vedant Bhagat, Sarah Costa, Oliver Green, Peiying Li, Forrester Welch

Northeastern University, Boston, MA, USA

Abstract

In continuation of the work on an NLP library with song lyrics data, we extract song data from the Spotify and Genius APIs. We build a library that loads the downloaded data, cleans and processes the data, and creates and runs Machine Learning models with kNN and Random Forest algorithms, attempting to predict/classify song genres from their lyrics. In data cleaning and processing, we remove annotations, punctuations, and stop words and perform lemmatization on the data. We generate n-grams and tf-idf vectorization for each song and use this data as inputs to train our machine learning models. We achieve moderate model accuracy, and future work on feature extraction and model tuning is needed.

Introduction

This project is a continuation of our team member Sarah Costa's work on an NLP library homework from DS3500 taught by Prof. John Rachlin at Northeastern University. Previously, song lyrics data was extracted using the Spotify API to gain insights via visualizations. To build on top of the previous effort in working with lyrics, we are interested to see if the lyrics of a song could be used to predict its genre. Therefore, our goal was to build and turn a machine learning model that could "accurately" predict a song's genre given its lyrics. Building an ML model to classify songs into genres based on their lyrics would be extremely useful for the industry. Spotify alone adds 60,000 songs to its library per day. We are interested in exploring ML methods that could be used to make it easier to classify songs. It would also be intriguing if we could deconstruct our results and make a song of a certain genre by using the most popular words that indicate that genre.

Methods

Data Acquisition

Previously, we extracted around 100 songs and their relevant information from Spotify. We then used the LyricsGenius library [1] to get the lyrics to the songs since this was not included in the Spotify data. First, we set up an access token with Genius so that we could make calls to the API. Then we looped through each of the songs that we had gotten data on from Spotify and searched the API for the lyrics that matched that song title and artist. We encountered "timeout" errors when trying to make consecutive API calls. We attempted to resolve this issue by using the sleep function from the time library and by troubleshooting through the LyricsGenius Github repository. In the end, we successfully solved this issue by initializing the Genius object (see driver.py line 22 for details) with a standard timeout of fifteen seconds.

After getting the song information from the Spotify API and the lyrics from the genius API, we compiled the information into one data frame. The data is then loaded into and stored in a pandas data frame called song_data in the lyric library (see LyricLibrary.py). The columns that we are working with include artists, genre, spotifyID, and lyrics.

Data Processing

The lyrics from lyric-genius API are cleaned in the clean_data method in the LyricLibrary. Annotations such as "[Chorus]" and "[Verse 1]" are removed from the string containing the lyrics of a song. Punctuation, digits, and stopwords are also removed from the lyrics. Similar words with different conjugations are lemmatized. Newlines are replaced with spaces, and the lyrics are saved as a string.

The song genres extracted from the Spotify API included many hyper-specific categories. To reduce the number of classifications the models would have to make, we consolidated genres into more general terms. A dictionary storing the mappings of the Spotify-defined genres to more general genres can be found in SpotifyScraper.py under general_genres. The generalize_genres method in LyricLibrary handles the conversion of specific to general genres.

Machine Learning Pipeline

We implemented two main functions for the machine learning pipeline to predict the song genres. The first method ML_pipline takes in a classifier, splits the data into training and testing sets, trains with the inputted classifier, then outputs a model accuracy score, which will be stored in our library. The second function run_all_models, serves as the main pipeline. This function takes in a dictionary of models with their model settings, for example, {"knn": [3,5,7,9,11], "Forest":[4,5,6]}, and will run all models with the configured settings. In the example dictionary, the numbers for "knn" are the number of neighbors, whereas the ones for "Forest" are the max_depths for the trees. The output model accuracies will also be stored in the lyric library.

In addition to the ML pipeline, we also implemented a visualization method to plot the accuracy scores as the model parameter changes. These visualizations will help us decide on the high-performing model parameters. Example plots will be shown in the analysis section below.

Analysis

The most important improvements to the models' accuracies came from an increase in training data and a consolidation of genres. While the effort to overcome the timeout error with lyric-genius was ongoing, we attempted to run the ML_pipeline with a limited dataset. This resulted in an overall accuracy of 1%, scarcely different from randomly guessing classifications. When we got through the error, we managed to download a full dataset with roughly 10,000 songs. After running the models with enough data, the accuracy jumped to around 9%. After implementing our consolidation of specific to general genres, the accuracy jumped to a range of 30-35%.

The K-nearest neighbor model increased the model accuracy by a small 1-2% after updating the distance function to use cosine distance. The cosine distance function is used because our tf-idf data is sparse (containing mostly 0s), and cosine has been shown to be effective for text analysis.

The run_all_models method in LyricLibrary can build and test k-nearest neighbor and random forest models with a list of given parameters. The plot_model_accuracy method creates two plots (shown below), one mapping max depth of random forest to accuracy and the other mapping number of neighbors to k-nearest neighbors accuracy. These plots show that the accuracy of a random forest model can slightly increase with a deeper max depth to an extent, and the accuracy of a k-nearest neighbor model can increase greatly until rounding off around the 15th neighbor.

The results do not signal great confidence in the ability to classify genre as a result of lyrics. There may be some correlation, but the subject matter of various genres has lots of overlap. To more accurately predict the genre of the song, we suggest investigating information about the beat, melody, and instruments.

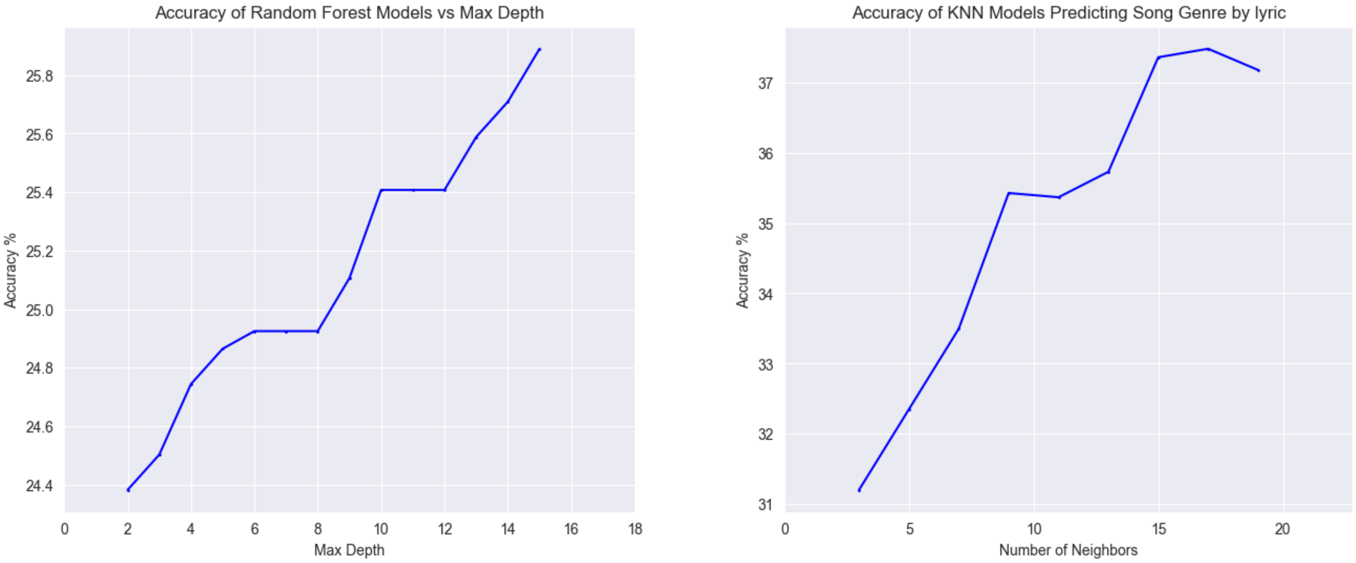

Fig 1., Fig 2. : Accuracy Scores of ML Models as the Max_Depth Changes

Conclusions

The lyrics of a given song can lend some insight into the song's genre, but not with great accuracy or reliability. Creating a data frame where every unique 2-gram or 3-gram is a feature of the dataset results in a very wide data frame with some 18,000 columns. Having only 10,000 songs in the dataset gives us a poor ratio of features to samples. To improve upon this, we suggest restructuring the lyric data to somehow shrink the number of features and significantly increase the number of elements in the data. However, the Genius lyric data does not contain nearly as many songs as Spotify. We also suggest looking into other song data (as features) besides lyrics to predict genre. Different genres of music have much in common in terms of lyrical content, but the musical feel of a song may be more defining.

Author Contributions

Each team member contributed equally to the coding, presentation, and drafting of this report. Sarah and Oliver worked on data acquisition, Forrester and Peiying worked on data cleaning and ML model building, and Vedant implemented the driver script to run the code. We all individually took part in the presentation and the written portions of the report.

References

[1] Johnwmillr. “Johnwmillr/Lyricsgenius: Download Song Lyrics and Metadata from Genius.com 🎶🎤.” *GitHub*, https://github.com/johnwmillr/LyricsGenius.

[2] Murthy S., A.R., Goodwin R., Keskinocak P. Rachlin J., Wu F., Yeh J., Fuhrer R., Kumaran S., Aggarwal A., Sturzenbecker M., Jayaraman R., Daigle R., *Cooperative Multiobjective Decision Support for the Paper Industry.* Interfaces, 1999. **29**(5): p. 5-30.

[3] Rachlin, J., et al., *Biological context networks: a mosaic view of the interactome.* Mol Syst Biol, 2006. **2**: p. 66.