CSE158 Assignment 2

Music Genre Classification From Lyrics

Jiyoung An
A16178228

Dongyoon Kim
A16080616

Jared Chen
A15908895

Ikjoon Park
A15894208

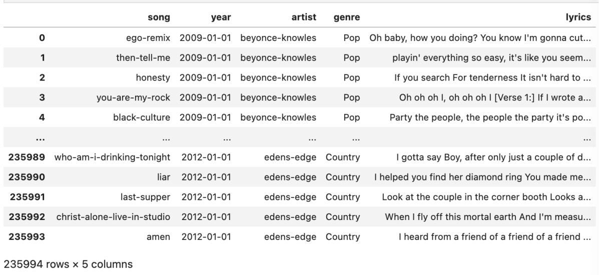
Abstract

This document is submitted as Assignment 2 for the CSE 158 Fall 2022 course. According to the 20 Trending U.S. Media and Entertainment Industry Statistics of 2022, the market size of the U.S. music industry is estimated at $43 billion, making it a highly competitive field.[1] To improve customer experience, music streaming services like Apple Music, Amazon Music, and Spotify have implemented advanced algorithms that use various features like search histories and listening histories to recommend music to their users. In this assignment, we attempted to develop a similar system by building our own music recommender that predicts the genre of a song based on its lyrics. The complete code used in this report is available at `https://github.com/cse158-fa22-team-pushystrokers/a2`.

Contents

1 Dataset and Phenomena

1.1 Dataset



	song	year	artist	genre	lyrics
0	ego-remix	2009-01-01	beyonce-knowles	Pop	Oh baby, how you doing? You know I'm gonna cut...
1	then-tell-me	2009-01-01	beyonce-knowles	Pop	playin' everything so easy, it's like you seem...
2	honesty	2009-01-01	beyonce-knowles	Pop	If you search For tenderness It isn't hard to ...
3	you-are-my-rock	2009-01-01	beyonce-knowles	Pop	Oh oh oh I, oh oh oh I [Verse 1:] If I wrote a...
4	black-culture	2009-01-01	beyonce-knowles	Pop	Party the people, the people the party it's po...
...	...	...	...	...	...
235989	who-am-i-drinking-tonight	2012-01-01	edens-edge	Country	I gotta say Boy, after only just a couple of d...
235990	liar	2012-01-01	edens-edge	Country	I helped you find her diamond ring You made me...
235991	last-supper	2012-01-01	edens-edge	Country	Look at the couple in the corner booth Looks a...
235992	christ-alone-live-in-studio	2012-01-01	edens-edge	Country	When I fly off this mortal earth And I'm measu...
235993	amen	2012-01-01	edens-edge	Country	I heard from a friend of a friend of a friend ...

235994 rows × 5 columns

Figure 1: original dataset

The raw dataset had 235,994 samples with 5 columns; song, year, artist, genre, and lyrics.

song name of the song

year the year of song published

artist artist name of the song

genre music genre of the song

lyrics lyrics of the song

[1] 20 TRENDING U.S. MEDIA AND ENTERTAINMENT INDUSTRY STATISTICS [2022] `https://www.zippia.com/advice/media-and-entertainment-industry-statistics`

In the raw dataset, the total number of songs was 235, 994, the same as the number of samples. Also, the total number of artists was 14, 139, and there were 12 genres.

1.2 Phenomena

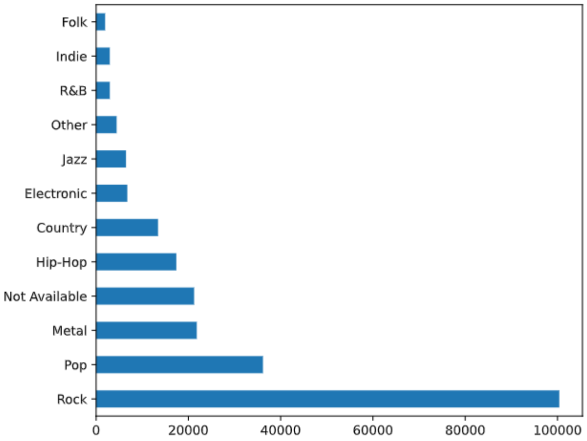


Figure 2: number of songs per genre

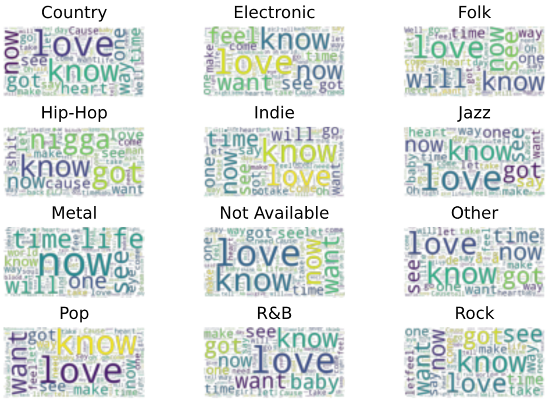


Figure 3: most common words in lyrics per genre

After preparing the data, we discovered several phenomena regarding lyrics and genre: (i) There were total 12 genres on this dataset, and the most popular genre is Rock (Figure 2). The interesting phenomena were that (ii) there are certain words that are found more often in each genre (Figure 3) and that (iii) the average length of lyrics vary depending on the genre. The genre that has the longest lyrics is Hip-Hop, while Metal has the shortest lyrics (Figure 4).

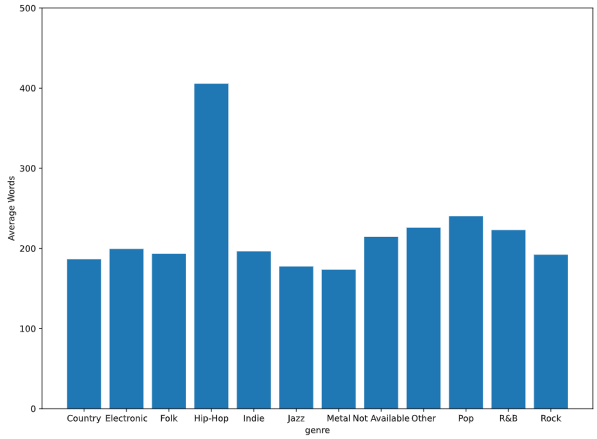


Figure 4: average number of words in lyrics by genre

2 Predictive Task

2.1 EDA

We pre-processed the data prior to attempting the predictive task. We attempted to exclude outliers from the samples and further processed certain data such as:

genre

cleaned **genre** by removing songs that had genres of **Not Available** or **Other**

lyrics

removed capitalization and punctuation

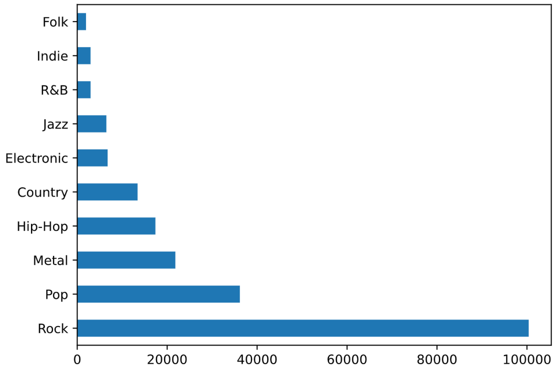


Figure 5: number of songs per genre after cleaning

After data processing, the number of genres have decreased from 12 to 10. We can now begin to create a predictive classification model for which songs belong to a genre based on its lyrical content. The number of songs per genre and the most common words per genre after processing data are shown in Figure 5.

2.2 Features and Label

Our first predictive task was described as

$$f(\text{artist}, \text{lyrics}) \rightarrow \text{genre}$$

However, using our baseline model, the score was almost 0.95 (`C=3.1622776601683795, class_weight =balanced, solver=saga;, score=0.949 total time =11.4min`) This is **too** high because of the `artist` feature was too closely correlated with the dependent variable *genre*. We realized that almost all artists likely only write songs in a genre they specialize in. This comes as no surprise, can you imagine Eminem singing a country song? Although this assumption was the likely culprit, we confirmed this by analyzing further.

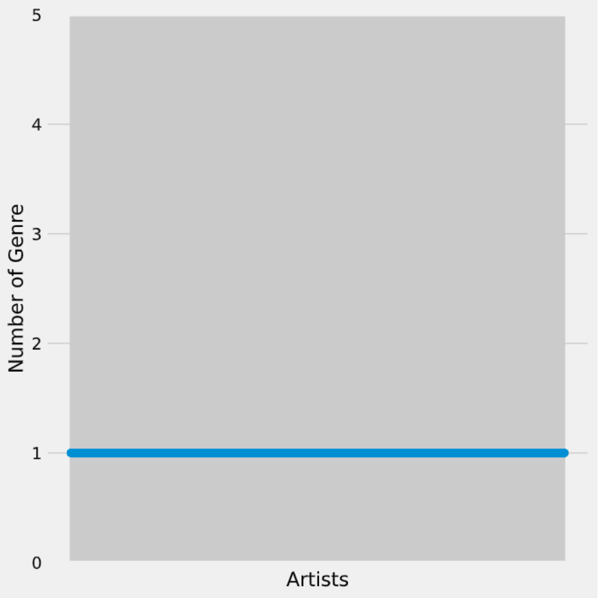


Figure 6: number of genres per artist

As shown in Figure 6, every artist is specialized in only one genre, which means if we use *artist* as part of our feature data, the predictive task becomes almost negligible. Thus, we decided it was correct to remove the feature *artist* from our predictive task. Our final predictive task can now be described as

$$f(\text{lyrics}) \rightarrow \text{genre}$$

2.3 Predictive Tasks Overview

Using Logistic regression model, the baseline model, our prediction accuracy was around **0.368** (`C=0.03162277660168379, class_weight=balanced, solver=saga;, score=0.368 total time= 1.3min`). We thought this was a reasonable basis to build from and worked to further develop our model. Through trial and error, and a switch in model choice, we achieved a final accuracy of **0.586** (`batch_size=1024, epochs=5;, score=0.586 total time= 3.1min`). The next section will discuss the models we used and how they worked.

3 Models

For this project, we ultimately used two models that we learned from class.

3.1 Logistic Regression (Baseline)

After processing the lyrics in the dataset, we are left with strings of length greater than 20 and void of punctuation and capitalization. We then tokenized these strings, resulting in a vector of words. We then removed the stop-words, common words that are deemed insignificant, from this vector to encourage the model to consider only important words. In an effort to better address the dynamic nature of natural languages and the many colorful ways artists may use different words, we stemmed and lemmatized our vector of words.

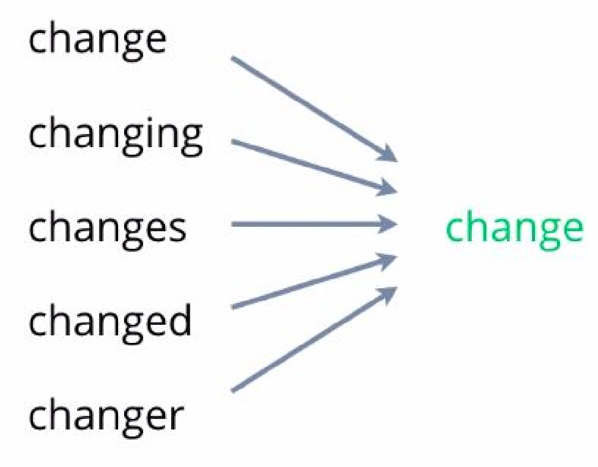


Figure 7: lemmatization of word ”change”

The goal of stemming and lemmatization is to reduce words to their ”root” or ”base” form, an important step towards text normalization. Stemming and lemmatization may also lessen the computational power needed to work on the dataset by allowing us to work with unique stems or roots rather than unique tokens.

Following these pre-processing techniques, we are able to extract a vector of unique token counts and a vector of token frequencies/weights. These weights were calculated using each token's TF-IDF, a measure that determines the significance of a word depending on its relative frequency within a song.

We used a multinomial logistic regression model to explore the predictive relationship between an artist's lyrics and the genre their songs belong to. We decided that a logistic regression model was appropriate because of our previous experience with it, its ability to be extended to multi-class classification, and its relatively fast training time. In addition, the data we are working with is not high-dimensional and thus helps avoid overfitting.

One way we optimized our logistic regression model is through K-fold cross-validation to avoid overfitting to the training set. This means that we divided the data set into k-folds and use 1 fold for testing and $k-1$ folds for training. We used grid search cross-validation (`sklearn.model_selection.GridSearchCV`, `CV` stands for cross-validation) in conjunction with this to tune our hyperparameters. Grid search cross-validation works by testing various hyperparameters, eventually returning the tuned hyperparameters and the associated accuracy.

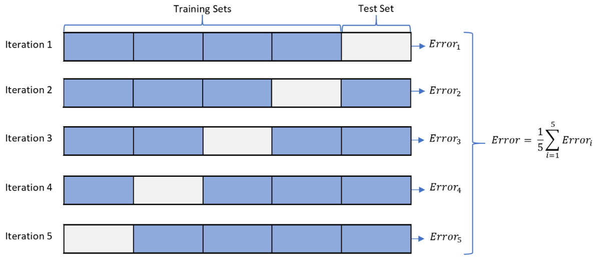


Figure 8: Cross Validation

3.2 LSTM

A more complex, more robust model alternative to logistic regression is a neural network. During the training of our baseline model, we found several limitations inherent to the linear model that have prevented us from reaching optimal accuracy. For instance, while tokenizing our sample texts, we did not take into account the linguistic contexts, and what's even worse on top of it we removed the stop words that could potentially help us identify the contexts (for example, the word "love" does not always have positive connotations; when prefixed with "don't," the phrase has a negative connotation). Although sentimental analysis through logistic regression is possible, we chose to not go down this route as it would most likely not significantly improve the performance of our model but further complicate our training process. An LSTM on the other hand, allows us to capture temporal features in our data; we hypothesized it would thus also capture connotations.

The neural network we used in this categorization task is basically an LSTM network (`keras.layers.LSTM`) sandwiched between several convolutions neural network (CNN) layers (`keras.layers.Conv1D`) and a fully-connected layer (`keras.layers.Dense`). Unlike our previous model, the network vectorizes the data by passing it through an embedding layer (`keras.layers.Embedding`), functionally a transformer, that encodes the texts by turning sentences into sequences of numbers thus preserving the linguistic flow of information.

To prevent over-fitting within the network, dropout layers (`keras.layers.Dropout`) were added between the CNNs and their connections to the LSTM so that only a certain number of weights (80%-90% as specified in our code) can be used to produce the final results; Outside the network, like in our previous model, stratified K-Fold cross-validation was used to maximize the amount of data fed to the model and prevent over-fitting as a result of focusing on a fixed portion of data for training or training with data containing the number of classes disproportional to the one in testing or validation.

4 Related Literature

Our dataset is from Kaggle[2]. We found another study case with a similar dataset as ours. This work is also from Kaggle [3] and was written by Reinhard Sellmair on November 2nd of 2019.

4.1 Other study case: Artist classification by song lyrics

This project dealt with the prediction of artists from lyrics. Here is the summary.

[2] https://www.kaggle.com/code/danofer/music-lyrics-clean-export

[3] https://www.kaggle.com/code/reisel/classification-of-artist-by-song-lyrics

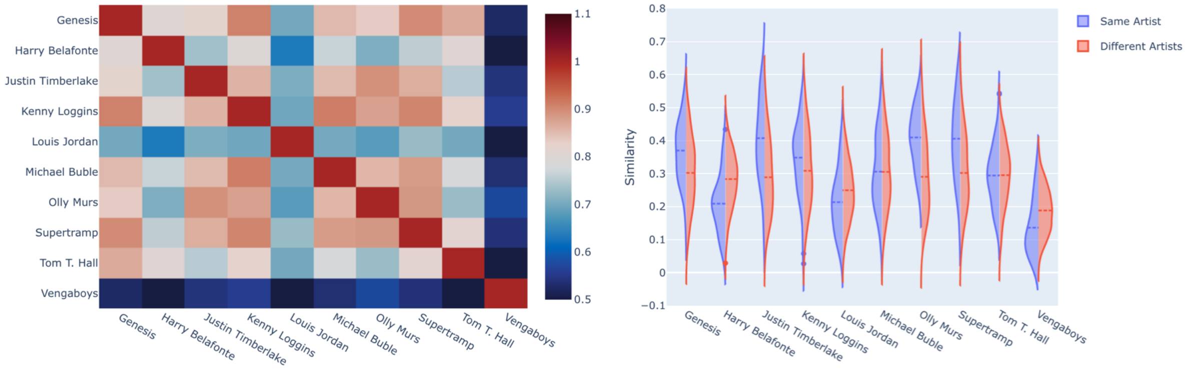


Figure 9: Similarity of artist and Similarity of songs

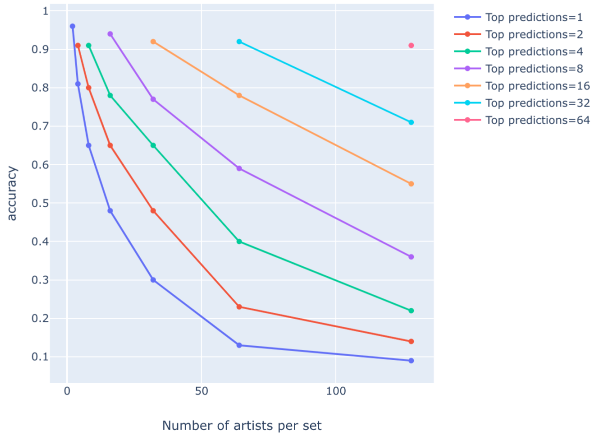


Figure 10: Accuracy of Sellmair work

I Pre-processing step & EDA

Similar to our work, Sellmair also removed unnecessary data, such as words in brackets, square brackets, line breaks, and non-English songs. He also tokenized and stemmed his data.

Furthermore, Sellmair handled duplicated songs and artists by removing them.

II Models

The models used in this project were TF-IDF and logistic regression. Using TF-IDF, Sellmair got TF-IDF scores per artist, vector similarity between each artist, and similarity of songs (Figure 9).

Additionally, Sellmair did sentiment analysis. This algorithm was also necessary for us to evaluate whether a certain word in lyrics had a positive or negative meaning when analyzing lyrics. Sellmair used the TextBlob library to implement this feature.

III Prediction

Using made models and various features, for example, the number of words, repeated words, word per line, and word frequency, Sellmair attempted to predict artists from song lyrics. Sellmair used half of the dataset as training data and the other half as validation data.

IV Validation

The accuracy that Sellmair achieved varied depending on the number of artists per set (Figure 10). This means that the accuracy increases as the number of samples (artist) decreases.

4.2 Compare and Enhance

The concept of analyzing lyrics and comparing words is very similar to ours, even though the prediction result is different than ours (artist vs genre). Thus, we found this study to be very relevant, useful, and interesting. In particular, we think that a sentiment language analysis performed by Sellmair could have further improved our own model and accuracy. Since even the same word could have completely different meanings depending on its polarity, we thought that this sentiment analysis should be applied to our model later.
Also, just as the accuracy was derived by manually adjusting the number of artists, we could manually adjust the number of genres, for example, by adjusting a specific label such as hip-hop vs. jazz, or hip-hop vs. rock, to improve our model a bit.

5 Evaluation

5.1 Results

Our initial attempt at training the logistic regression model yielded an accuracy of approximately 0.368, which was unsatisfactory. We later discovered that we had removed the "Not Available" category from some of the songs in the dataset but overlooked removing the "Other" category as well. Although this resulted in a slight improvement in our model's accuracy, by only 0.01s (from 0.368 to 0.400), we were still not satisfied with the final result. Recognizing that the preservation of linguistic flow could be crucial to genre prediction, we decided to abandon the logistic regression model and explore a new approach using an LSTM neural network, as discussed in section 3.2. This change resulted in a significantly improved accuracy of around 0.586, marking a notable improvement from our initial attempt.

5.2 Conclusion

Upon initial data preprocessing, our model's accuracy was 0.386. However, after implementing a more comprehensive data cleaning and normalization process, as well as utilizing an LSTM neural network model, the accuracy increased to 0.586. While we had hoped for an even higher accuracy, we acknowledge that predicting music genre from lyrics is a complex task that requires significant computational resources.

During the development process, we encountered a significant challenge in identifying word-genre overlap, where certain words or phrases commonly used in one genre can also appear in another genre but with different meanings. For example, the word "love" may appear in both hip-hop and RB songs, but with vastly different connotations and contexts. As a result, our model struggled to accurately classify such songs into their appropriate genre, often misclassifying them as pop, RB, or jazz instead of hip-hop.

To further improve our model's accuracy, we have identified several potential strategies. One approach is to develop features that count the most common words in each genre and assign them a weight in the genre classification process, enabling the model to better distinguish between genres with similar vocabularies. Another strategy is to incorporate techniques from existing literature, such as Sellmair's method of removing non-English songs, square brackets, and curly brackets from lyrics. We believe that implementing these techniques and exploring additional optimization methods could further improve the accuracy of our model in future iterations.