CSE158 Assignment 2

Music Genre Classification From Lyrics

Dongyoon Kim
A16080616

Ikjoon Park
A15894208

Jared Chen
A15908895

Jiyoung An
A16178228

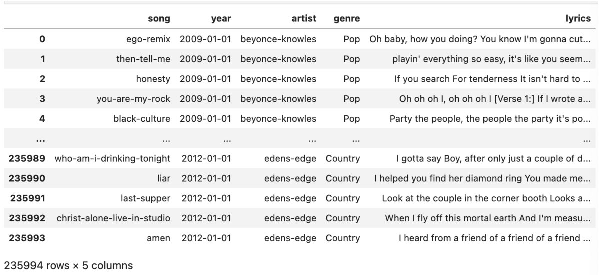
Abstract

This report is for Assignment 2 of CSE 158 Fall 2022 class. As of 2022, the U.S. Music industry has a market size of $43 billion.[1] Therefore, music streaming services, such as Apple Music, Amazon Music, and Spotify, try to heighten the customers service experience. They've developed advanced algorithms to recommend music to their users based on various features, such as users' search histories and listening histories. We have attempted to mimic this by building our own recommender. Our primary goal in this assignment is the successful prediction of song genre from its lyrics. The code used in this report can be found here `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

In the raw dataset, the total number of songs was $235,994$, the same as the number of samples. Also,

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

the total number of artists was 14, 139, and there were 12 genres.

1.2 Phenomena

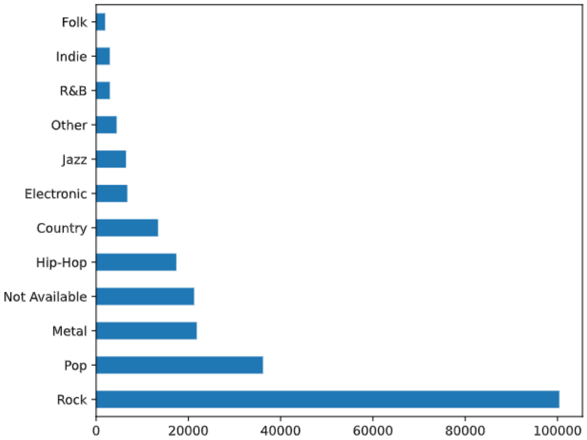


Figure 2: number of songs per genre

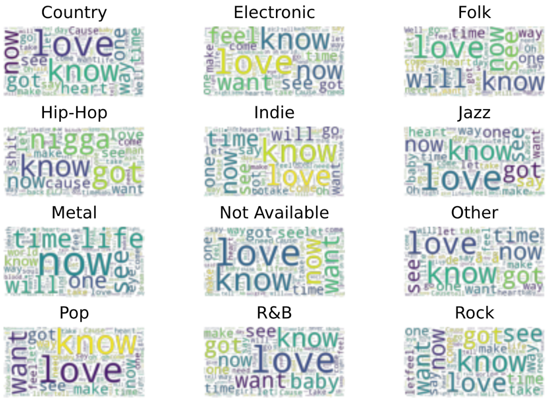

Country
Electronic
Folk
Hip-Hop
Indie
Jazz
Metal
Not Available
Other
Pop
R&B
Rock

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).

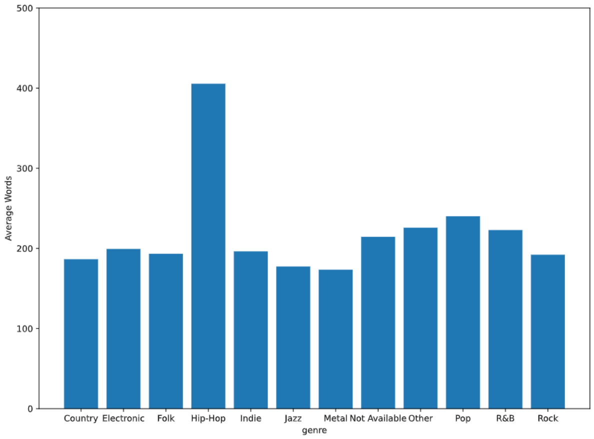


Figure 4: average number of words in lyrics by genre

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

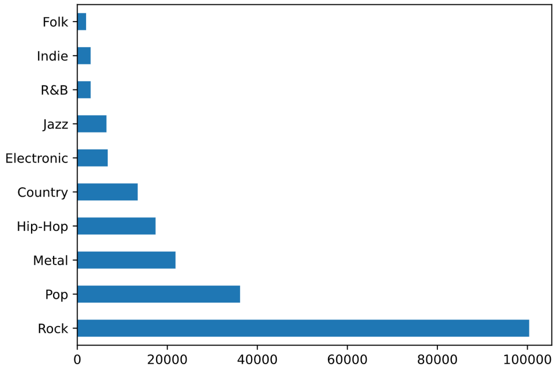


Figure 5: number of songs per genre after cleaning

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 3.

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.

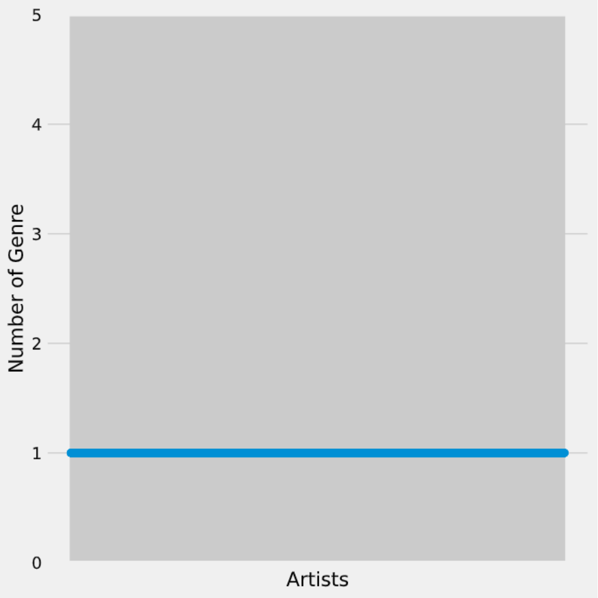


Figure 6: number of genres per artist

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.

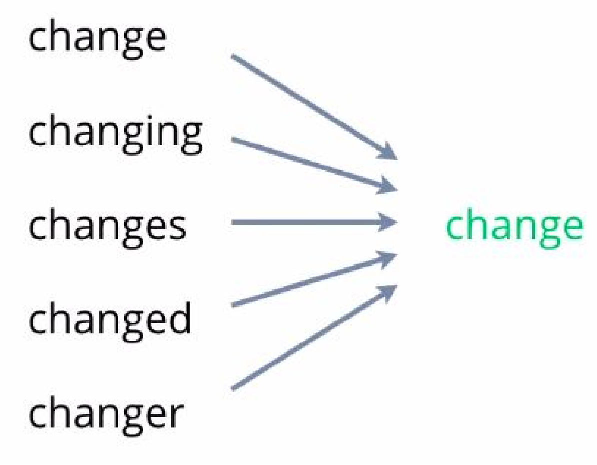


Figure 7: lemmatization of word ”change”

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.

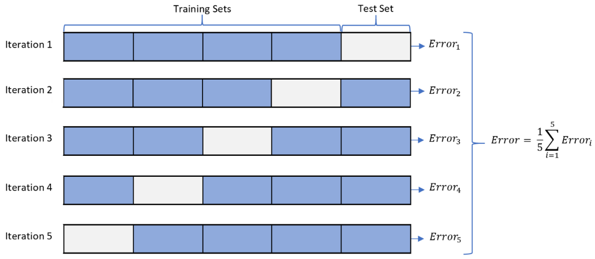


Figure 8: Cross Validation

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.

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.

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

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

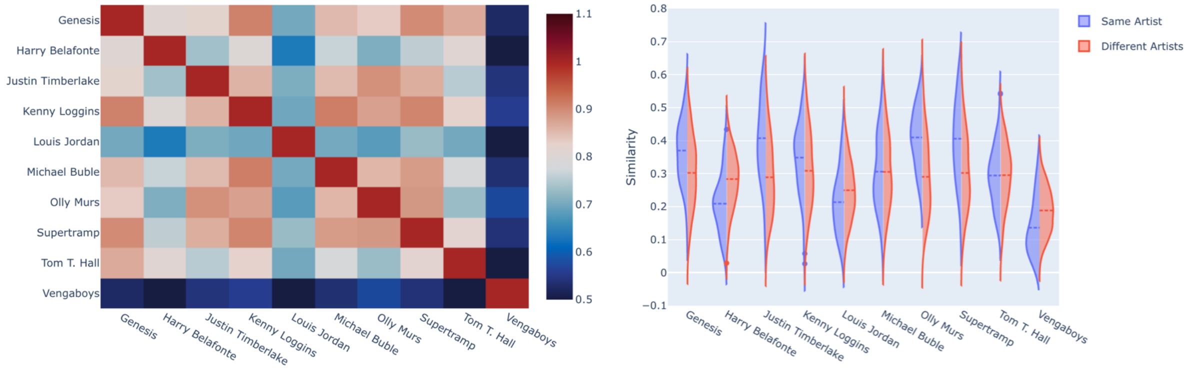


Figure 9: Similarity of artist and Similarity of songs

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 (Figure 8), and similarity of songs (Figure 8).

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.

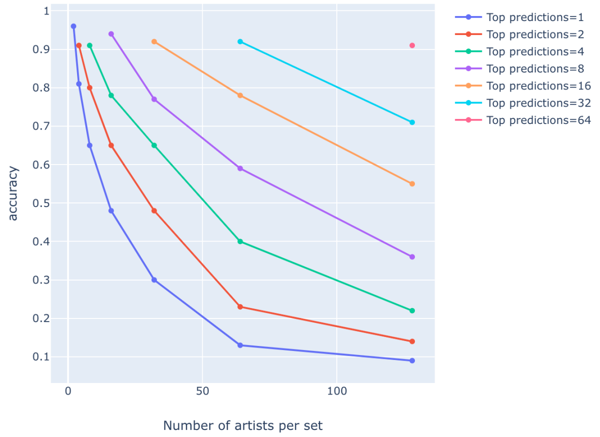


Figure 10: Accuracy of Sellmair work

IV Validation

The accuracy that Sellmair achieved varied depending on the number of artists per set (Figure 9). 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 our logistic regression model was quite unsatisfactory and only yielded an accuracy of around 0.368. In an effort to improve our accuracy, we realized that we removed the "Not Available" category that some of the songs in the dataset had, but failed to realize that the "Other" category should be removed too. Although this change succeeded in marginally improving the accuracy of our logistic model, by a small magnitude of 0.01s or so (from 0.368 to 0.400), we were still left dissatisfied with the final accuracy. We suspected that a key to genre prediction lay in the preservation of linguistic flow, rather than the simple tokenization and lemmatization/stemming of words. To address this, we abandoned our logistic regression model in favor of an LSTM neural network as discussed in section 3.2. This shift led to a significantly improved accuracy result, an increase from 0.368 to around 0.586.

5.2 Conclusion

Our accuracy was 0.386, prior to data preprocessing. However, after cleaning, normalizing, and removing unnecessary data, and changing our model to an LSTM neural network, the accuracy was increased to 0.586. Although we expected a higher accuracy and it was clear that the model could be further optimized, we realized that it is hard to predict music genre from lyrics due to the limit of our computational resources. We also found that a major obstacle was word-genre overlap despite differing contexts. For example, some songs mentioned love in the hip-hop genre while others only mentioned things like money and other topics. Thus, our model had trouble identifying these songs as belonging to the hip-hop genre, and instead tended to attribute to the pop, R&B, or Jazz genre.

To further increase our accuracy, we could build features that count the most common words and give them some points, so we can use them when we calculate the score when there are genres that have a similar probability. Also, we could take a page from Sellmair and remove non-English songs, square brackets, curly brackets, etc from the lyrics to increase predictive accuracy.