University of Victoria
Department of Computer Science

Determining Genre By Song Lyrics

Summer 2023

CSC 503/SENG 474

By
Opara Anita [V00996592] Sebastien Perre [V00956188]
Performed at: University of Victoria, Victoria

Contents

1 Introduction

Suppose we have a bunch of song lyrics and we want to classify them into different genres. We will attempt to do this through various methods and find an optimal method for classifying them. We will get our data from genius's API, visualize it with various techniques such as word clouds, preprocess it and use several machine learning methods to create a model that has an accuracy of around 92 percent! Of course, to achieve this accuracy we went through many hurdles and we will explain how we managed them.

2 Getting the Data

NOTE: THE NOTEBOOK MUST RUN ON THE LOCAL MACHINE TO WORK

2.1 How it works

The data collection code starts off with using the genius charts and names of songs (entered manually), that are in two different genres. In this case, we chose country and rap due to how different in style they are. Next, we request info about the genius songs in the charts. Furthermore, we use the URL and name of the song given in the info. We use the name of the song to see if there are any duplicates in the names of the songs we added manually so there isn't any duplicate data. Then, we make API requests to get the lyrics of each song and store it in a dataframe consisting of its lyrics and the genre its in. Finally, we save this dataframe to a CSV file called data.csv to use for machine learning.

2.2 Problems with virtual machines and genius's API

Our first problem we encountered was with getting requests through Genius's API in a virtual machine. It seems this problem is due to Genius tightening their CAPTCHA rules and as a result people that run code from virtual machines get timed out [1]. Thus, we had to run the data collection notebook on a local machine in order for it to work.

2.3 Timeout issue with API Requests

The second problem is that when we tried to request song lyrics from the Genius API, it would timeout. This is due to the fact that the Genius API can only handle so many requests at a time. To bypass this problem, we made a request and if it failed we would try again until it was successful.

2.4 Not enough songs in the charts

In the Genius charts, there are only 128 songs for country and 200 songs for rap which isn't enough data to make a good model. Thus, we had to manually add song names and make sure we didn't add the same song twice. We added song names until we had 250 songs for country and 250 songs for rap.

2.5 Artifacts in the data

In the lyrics data, there are artifacts that aren't part of the actually song lyrics such as the number of contributors that worked on transcribing the song lyrics, the different language translations of the lyrics, and the embeds. There are no options to remove these artifacts and they don't seem to affect the model's accuracy so we just decided to leave them since removing them would be tedious and take time.

2.6 Getting songs from all five genres

Get 100 songs from each genre available to us in the charts (pop, rap, rb, country, and rock) using the same methods as above, but we didn't have to manually add songs due to the charts having 100 or more songs in them.

2.7 Code

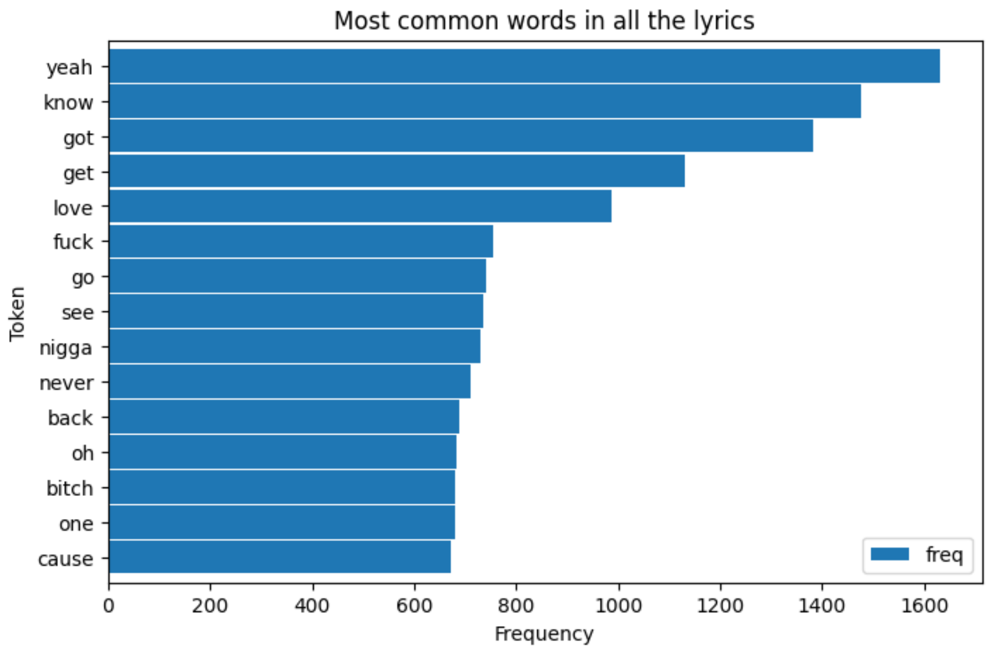
Go to data_collection.ipynb to see the code

3 Visualization

All the visualisations were created using tokens (keywords) generated from the pipeline named pipeline_sk. This pipeline makes text lowercase, tokenizes, removes english stopwords and non english words from the generated lyrics to ensure only keywords are used for analysis.

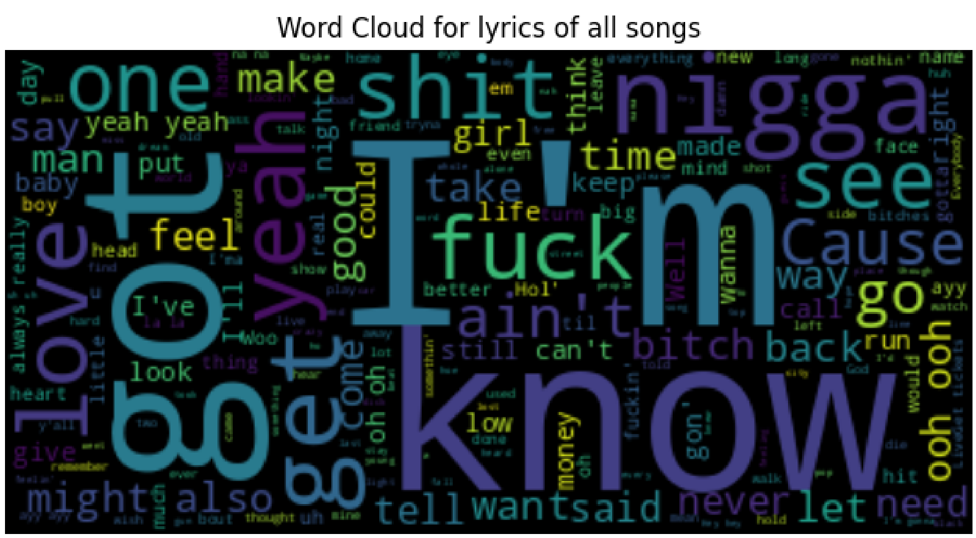
3.1 Most Common Words In All The Lyrics

This barchart highlights the top 15 words that occurred in the entire dataset and their relative frequency. We identified 'yeah' as the most common word in the dataset.

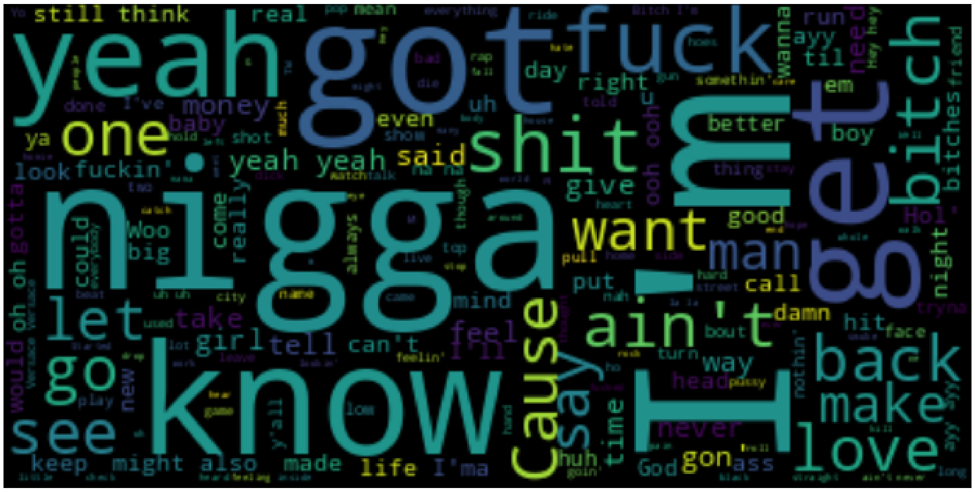


3.2 Word Cloud

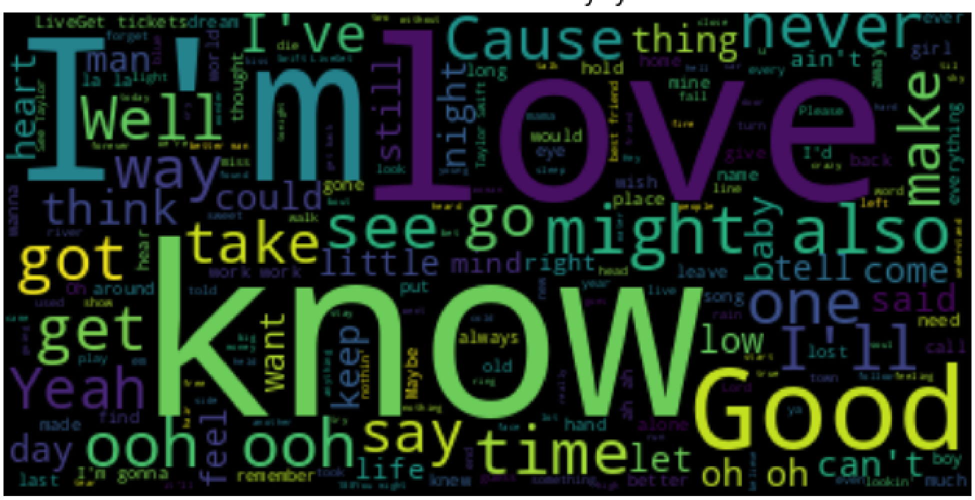
The word cloud shows a textual representation of tokens identified from the dataset where the size of the word corresponds to the importance of the words in the dataset. The word cloud generated uses the top 200 tokens from the specified lyrics. As we can see in the word clouds, rap songs have more vulgar language while country songs have more formal language.



Word Cloud for rap lyrics



Word Cloud for country lyrics



3.3 Sentiment Analysis

Sentimental analysis is a natural language processing technique used to identify emotions and tones in words. We used sentiment analysis to check out the tone portrayed in lyrics of our data set. We calculated the sentiment score for each song and calculated the mean of all the sentiment scores in each genre. Country lyrics had a score of 0.110879 and Rap lyrics had a score of 0.00409. Country lyrics has approximately 27 times better sentiment compared to Rap Lyrics.

3.4 Code

Go to Visulalizations.ipynb to see the code

4 Models

4.1 Preprocessing

To preprocess the data, we split the data into features and labels followed by spliting the data in a 80/20 training-test split. Next, we made the lyrics all lowercase and passed it into a TfidfVectorizer with various stopwords.

4.2 First Model Iterations

We decided to create two start models, an SVC model and a Multinomial Naive Bayes model with default settings. These two models did fairly well as first iteration models with the SVC model scoring an accuracy of 88 percent and the Multinomial Naive Bayes model scoring an accuracy of 86 percent. Figure 1 a) and b) shows the classification report of both models.

	precision	recall	f1-score	support
country	0.85	0.92	0.88	50
rap	0.91	0.84	0.87	50
accuracy			0.88	100
macro avg	0.88	0.88	0.88	100
weighted avg	0.88	0.88	0.88	100

(a) SVC

	precision	recall	f1-score	support
country	1.00	0.72	0.84	50
rap	0.78	1.00	0.88	50
accuracy			0.86	100
macro avg	0.89	0.86	0.86	100
weighted avg	0.89	0.86	0.86	100

(b) Multinomial Naive Bayes

Figure 1: Classification Reports of First Models

4.3 Finding the Best Parameters

Now that we have our first models, we fine tune them by doing grid search with multiple parameters and getting out the accuracies and best parameters to determine our best and final model. Doing grid search on SVC gave us an accuracy of around 90 percent and Multinomial Naive Bayes gave an accuracy of around 92 percent.

4.4 Final Model

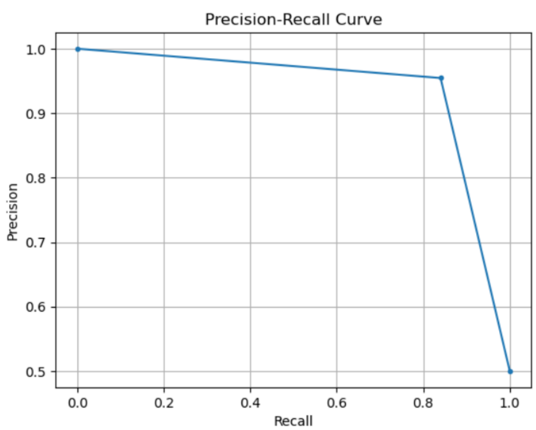
Since Multinomial Naive Bayes gave the best accuracy out of all, we choose it to be our final model. Thus Multinomial Naive Bayes with idf off, alpha of 0.01 and fit prior being false gave our best model. Figure 2 shows our classification report of this model.

	precision	recall	f1-score	support
country	0.95	0.84	0.89	50
rap	0.86	0.96	0.91	50
accuracy			0.90	100
macro avg	0.91	0.90	0.90	100
weighted avg	0.91	0.90	0.90	100

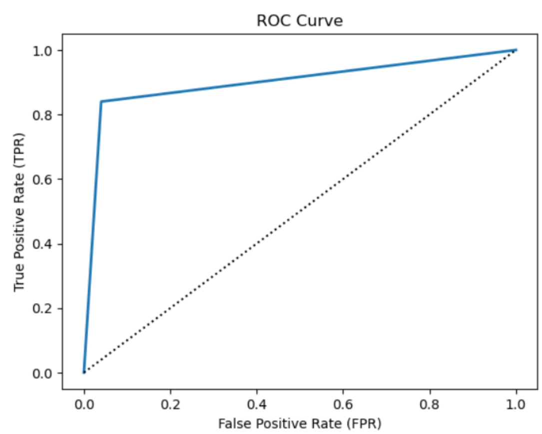
Figure 2: Classification Report of our Final Model

4.5 Model Analysis

We analyze our model with a Precision-Recall Curve, ROC Curve and a Confusion Matrix. Looking at the Precision-Recall (Figure 3 a) and ROC Curves (Figure 3 b), we see that our model did very well. Though as we can see in all three graphs, it seems that precision for country is higher than recall for country which means that they classify more country songs as rap than rap songs as country songs. A learning curve shows the relationship between the test accuracy and train accuracy. The model's learning curve in Figure 5 shows our model does not over fit the data as our test accuracy is about 92



(a) Precision-Recall



(b) ROC

Figure 3: Precision-Recall and ROC Curves

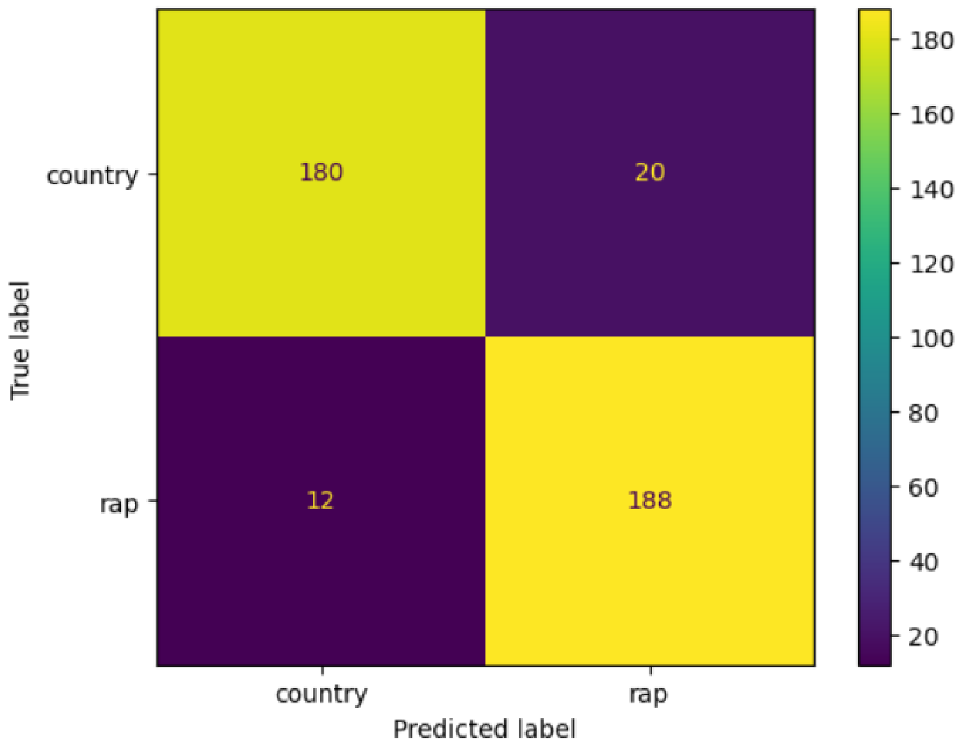


Figure 4: Confusion Matrix

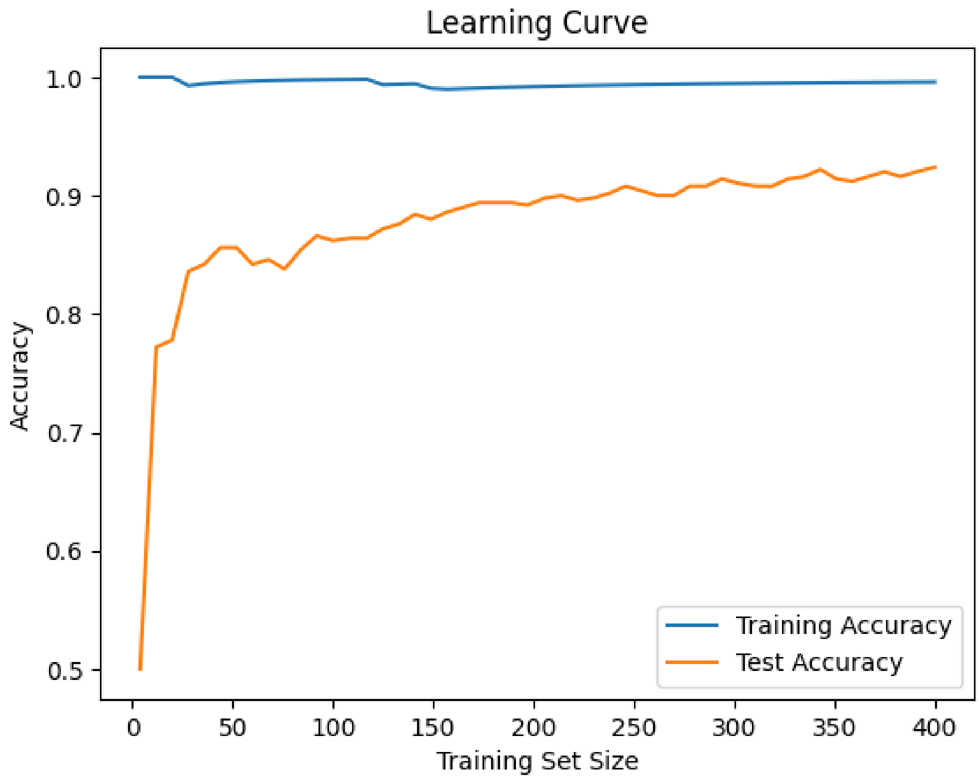


Figure 5: Learning Curve

4.6 Multi-classification Model

We tried doing a multiclass classification with data consisting of 500 songs where there was 100 songs from each genre (pop, rap, rb, rock, country) though the model did terrible due to songs being multiple genres. Thus, we decided to scrap the idea for now, but it would be a good followup project.

4.7 Code

Go to models.ipynb to see the code

5 Improvements That Can Be Made

5.1 Clean Up Data

Our data had lots of artifacts, it didn't seem to affect the accuracy of the model that much, but if removing it caused a 1-2 percent increase it would be a worthwhile endeavor.

5.2 More Genres

An improvement to the model would be the ability to accurately classify songs of other genres.As this would make the classifier more widely used.Adding rock songs would probably well with our classifier as rock,rap and country lyrics are very distinct and have very few overlapping songs.Challenges might arise when dealing with pop and r&b songs as they are have very similar lyrics and a lot of overlap.

5.3 Multi-Label Classification

Our model was based on the idea that one song belongs to one class. A great addition would be the ability for the classifier to be able to identify all the genres a particular song belongs to as a song can belong to multiple genres.

6 Conclusion

In this project, we were able to collect lyrics for rap and country songs from the genius API, perform analysis on it to identify key findings, and develop a well tuned model to classify rap and country genres with the Multinomial Naive Bayes approach with high accuracy.

References

[1] "Getting 403 error on vps," [Online]. Available: `https://github.com/johnwmillr/LyricsGenius/issues/220`. [Accessed 20 June 2023].

[2] "Genius Docs," [Online]. Available: `https://lyricsgenius.readthedocs.io/en/master/reference/genius.html`. [Accessed 14 June 2023].

[3] "Getting Lyrics of Songs via Python (LyricsGenius)," [Online]. Available: https://medium.com/analytics-vidhya/getting-lyrics-of-songs-via-python-lyricsgenius-23e5dd5992e3. [Accessed 14 June 2023].