# Genre Classification using Lyrics

**Ribhi El-Zaru, Luis Grisanti, Artem Shkuratov**

## Abstract

In this study we set out to evaluate the viability of lyrical features as a means of genre classification for contemporary songs. We obtain lyrics and information about 7383 songs from the Rap Genius and Spotify APIs, and compare three separate methods for classification: Naive Bayes machine using n-grams, advanced Naive Bayes machine with various added features, and a word vectorization model. Our findings indicate that lyrical genre classification can be successful significantly above the random baseline, with the advaced Naive Bayes machine producing the best results of our three methods.

## 1 Introduction

The composition of modern musical pieces can be summarized by two interplaying components: auditory and lyrical information. Each musical piece is categorized as a genre, presumably based on a shared set of musical conventions. When left to the individual listener, the process of genre-classification naturally relies mostly on auditory information. Auditory cues pointing to a particular genre are generally more noticeable to humans than lyrical features; often resulting in a trained listener's ability to deduce the genre of a piece before any lyrical components are introduced. In this study, we will attempt to establish whether lyrical information can be effectively used as a feature for genre classification through computational models. Upon completion of the model, we expect the results to affirm the viability of lyrical information as a means of genre classification. Depending on the results, this study may shed light on musical genres, which are typically taken to be determined by a piece's auditory experience. Thus, our findings hope to present a new, alternate method of automatic genre classification for musical data, one that takes advantage of the relatively small storage space required for lyrics as compared to auditory elements.

## 2 Related Work

There has been a significant amount of research dedicated to the study of lyrical genre classification in recent years. In the following section, we discuss the topics and methods utilized by three papers central to our work, and how each of their results influenced our approach to design and evaluate our classification techniques.

### 2.1 Rhyme and Style Features For Musical Genre Classification By Song Lyrics

In this paper, Mayer et al. (2008) present a novel set of features developed from textual analysis of song lyrics, and compare them to classical bag-of-words indexing approaches. They stress the importance of text pre-processing when utilizing lyrical collections, as the sets of features they opt to analyze (namely bag-of-words, rhymes, part-of-speech, and text statistic features) cannot be constructed from unstructured text. By pre-processing lyrics to understand these elements of any song, Mayer et al. were able to utilize these features to better their Naive Bayes, k-NN, Decision Tree and SVM approaches genre classification. However, their project suffers from the lack of data, as they only had 397 songs to work with. With our project, we hope to apply these pre-processed attributes to our database, and see how factoring for those in our models effects our classification accuracy.

### 2.2 Lyrics-based Analysis and Classification of Music

In this paper, Fell et al. (2014) experiment with various song features such as vocabulary, style, semantics, orientation and song structure to detect genre, distinguish song quality, and determine publication time of a song. Since no large lyric

datasets were made publicly available, Fell et. al opt to collect their own lyrics by scraping lyrics from the website Lyricsmode.com. After gathering songs, cleaning the data, and normalizing the notation style of these lyrics, they successfully build a final corpus of 400k English song texts of 7.2k unique artists. They decide to POS tag and chunk the lyrics in order to build a dataset pertaining to feature collection. After gathering the corpus, they build various features to be used to measure vocabulary richness, syntactic structure and style, egocentricity and repetition to assist in their models. With our project, we wanted to utilize similar feature to classify genre and estimate song-quality as well. Furthermore, since this project focused on artist ratings as a metric for popularity rather than song popularity, we figured that their classification of popularity would be askew, and hope to improve on this by using song popularity to train our model instead.

### 2.3 Lyrics-Based Genre Classification Using Variant tf-idf Weighting Schemes

In this paper, Ying et al. (2014) utilize a collection of 1000 English songs representing either Pop, Blues, Country, Folk, R&B, Reggae, Grunge, Punk Rock, Soul or Metal to study the correlation between genre and mood in order to improve lyrical genre classification. To determine song mood, the authors manually categorize the songs as either happy, sad, angry, relaxed, calm, gloomy, romantic, confident, disgusted or aggressive. After compiling the data, they used the WEKA Machine Learning toolkit to construct k-Nearest-Neighbors, Naive Bayes and SVM models to determine correlation between genre and mood. What caught our attention from this study is the employment and assessment of various weighting schemes used to introduce lyrical and mood information in the models. The authors use five variations of the popular tf-idf weighting scheme, three of which were composed of strictly lyrical features while the other two incorporated both mood and lyrical features. The results illustrate the significance of the impact that different weighting schemes can have on model performance, with results ranging from 70.97-79.14 (Lwf-idf) to 62.29-75.48 (wf-idf) accross each genre. Although we do not utilize the aforementioned weighting schemes in our model design, we have identified this technique as an interesting area of further research for our project.

## 3 Data

Being that a preprocessed dataset of songs with their lyrics and meta-data relevant to this study is sparse, we generated our own dataset utilizing the Rap Genius API and the Spotipy API as well as a list of 1001 albums along with their respective artist. After getting an album and artist, we utilized Spotipy to get the songlist of said album and the genres associated with said artist. If one of the genres of said artist fell into the genres we were working to identify – Folk, Rap, Rock, R&B, Country and Blues – we would add the songs in the album to our dataset. After getting the songs, we used Rap Genius' API to get a URL for the lyrics page and then utilized a web-scraper to pull the lyrics from the URL and Spotipy's API to get popularity and duration from the song. Then, we created a Song object using python out of these attributes. The Song object has the following attributes.

- Lyrics : lyrics of the song pulled from the web-scraper.
- Title: Name of the song
- Artist: Artist/Composer of the song
- Genres: Genres associated with the song.
- Popularity: Popularity of the song as indicated by Spotipy.
- Duration_ms: Length of the song in milliseconds.
- numVerses: Number of verses in the song.
- numChoruses: Number of Choruses in the song
- tokenizedSentences: The lyrics converted to a list of tokenized lists where each tokenized list was a tokenized line in the song.

At the conclusion of our data collection, we were able to collect 7383 songs from these 1001 albums. The distribution of these songs can be found in Table I.

### 3.1 Generating the Training and Test Data

When running our results with all of the songs we had, we noticed that our results were heavily skewed to rock due to the fact that it made up a large portion of our data. We believe that overfitting occurred and that the algorithms were likely to choose rock every time since it made up the majority of the data. This caused our results to be less focused on the lyrics and more focused on the probability of each genre occurring. At this point we decided we needed to equalize the amount of songs from each genre so that the classification came from lyrics instead of song distribution. In order to have equalize the amount of songs from

each genre, our data was reduced to 130 songs from each genre. By doing this, we were able to reduce the massive impact rock would have on our data set by making up 62% of all of our data. With a total of 780 songs, 80% from each genre became the training set and 20% from each genre became the testing set. This ensured that our training and testing sets were good models of each other and would not skew the results.

## 4 Methodology

For our genre classification task, we applied 3 distinct approaches to determine which approach would best classify genre given lyrics.

### 4.1 Naive Bayesian Inference using n-grams

We will build on our unigram Naive Bayesian Inference with n-grams to see if understanding recent context (n will be 2, 3, or 4 given that some sentences in lyrics are rather short) will be an improvement of simple word-frequency distributions.

### 4.2 Extending n-gram Naive Bayesian Inference by Adding Feature Analysis

We will try to improve on our n-gram based Bayesian inference by adding various lyric features to assist the model in predicting. These features are song length, # of lyrics per line, vocab richness (type-token ratio), egocentricity (measured by comparing the amount of first-person references to third-person references), and repetition.

### 4.3 Vectorization

We will also do a variant of neural word embeddings. We shall take all of the lyrics from our training set and combine them into a word tokenized list. On this list we will build a word2vec model in order to be able to represent every word as a vector. After this model is built, we create 6 lists where each list has all of the tokenized words for that genre. Using these lists and the model, we will vectorize every single word and add the vectors together and then divide the vector by the total amount of words used. This will result in a vector that should be the average word for that genre of music. This will result in 6 vectors, one for each genre, that can be used to figure out what genre an individual song is. When trying to learn the genre of a single song: we will word tokenize its lyrics, turn every word into a vector, add together each vector, and then divide that vector by the total amount of words. Once we have the average vector for that song, we should calculate this vectors cosine similarity to the 6 vectors for each genre. We feel that the vector that has the largest similarity is most likely to be the same genre of music.

## 5 Results

For our experiments, we calculated the probability of success as the number of genres the model predicted correctly divided by the total number of songs. We ultimately compared this probability of success against 2 baseline measures detailed in section 5.1. Furthermore, for our Naive Bayes experiments, we computed a confusion matrix to determine metrics of error, accuracy, recall, specificity, precision and F1-score for each genre that could be classified.

### 5.1 Baseline Comparisons

#### 5.1.1 Random Chance of Success

To evaluate our results, we compared all of our results to the random binomial baseline measuring the chance you guessing the genre of a song correctly completely at random. This was equivalent to $0.16666$ as we classified 6 potential genres.

#### 5.1.2 Human Chance of Success

We also compared our results against a human baseline that we calculated through a 12 question survey with 44 respondents. After collecting the results of this survey, we generated a 95% confidence interval for the probability that a human would successfully identify a song's genre given it's lyrics with the following equation:

k = # of correct genre guesses = 304
n = total # of guesses = 528
$\alpha = .05$
$100(1-\alpha)\%$ CI =

$$[k/n - z_{\alpha/2} * \sqrt[2]{\frac{(k/n)*(1-k/n)}{n}},$$
$$k/n + z_{\alpha/2} * \sqrt[2]{\frac{(k/n)*(1-k/n)}{n}}]$$

After amassing our data, we ended up with the following confidence interval.

$$[0.53360, 0.61791]$$

Thus, if our model was able to perform at an accuracy rate within the bounds of this confidence interval, we could say with 95% certainty that it lied in the same region of the true-ability humans have to correctly categorize genre.

Table 1: Basic Naive Bayes Results

| n-gram Size | Success Rate |
|---|---|
| 1 | 16.58% |
| 2 | 17.42% |
| 3 | 19.53% |

### 5.2 Basic Naive Bayesian Inference Results

#### 5.2.1 Results from Basic Naive Bayes

When viewing these results it is clear that they were not far from random baseline. However, it is interesting to see that as the size of the n-gram increases so do the results. Although the increases are small, they are not insignificant. An n-gram of size 1 hovers right around baseline while one of size 3 increases accuracy by close to 20%. Thus, we can conclude that word relationships in music are more important to genre classification than simple word frequency rates.

### 5.3 Advanced Naive Bayesian Inference Results

Table 2: Advanced Naive Bayes Results

| n-gram Size | Success Rate |
|---|---|
| 1 | 27.36% |
| 2 | 31.05% |
| 3 | 30.60% |

#### 5.3.1 Results from Advanced Naive Bayes

The Advanced Naive Bayes model differs from the Basic Naive Bayes model as we introduced the following features: words per minute, number of choruses, number of verses, and lines within the song. The results from the Advanced Naive Bayes were much better than the Basic Naive Bayes. With over a 10% increase it is clear that adding the features had a positive impact on correctly classifying genres. A surprising difference from the Basic Naive Bayes is that increasing the size of the n-gram does not always improve the results. The best results came from n-grams of size 2 as seen in Table 2. The n-gram of size 3 slightly decreased the results, which surprised us given the increase in the respective Basic Naive Bayes model.

### 5.4 Comparing the Basic Naive Bayesian Inference and the Advanced Naive Bayesian Inference

After constructing the two versions of the Naive Bayes models, we were interesting in seeing if we could statistically prove that the Advanced Naive Bayes was an improvement over the Basic Naive Bayes model. We did this by performing the non-parametric sign test on the two experiments. This was done by running the two models adjacently for 50 iterations, with a new training and test set each time, and comparing the accuracy of the two models to build the Sign Test test-statistic. After doing this, the Advanced Naive Bayesian model outperformed the Basic Naive Bayesian model every time, showing that the variance in the accuracy of the two tests was not enough to allow the Basic Naive Bayesian model to outperform the Advanced Naive Bayesian model. After performing the sign test, we could say that the p-value of the test was minute – about 3.874 to the power of $10^{-11}$– meaning that the Advanced Naive Bayesian model could be considered more effective with 99.9% confidence.

### 5.5 Vectorization

While it was assumed this would be one of our most successful results, it turned out the vectorization of the lyrics was not a suitable method of classifying the genre. The overall accuracy came to be 15.3% for our test data. This result was less than the baseline. We believed that since this method would generalize each genre down to a "one word vector", that these vectors would be significantly different from each other. Unfortunately, we found that this method led to over-generalization and the genre vectors were far too similar to each other. As seen in Table 8, the cosine-distance between any of the genres were very small. This small distance between the vectors showcased just how similar they were and how ineffective they would be. This was due to the fact that adding together thousands of vectors ended up averaging itself out. Since these results were so inconclusive, we decided to build word-clouds for the lyrics of each genre. These word-clouds revealed that many of the genres actually had very similar word distributions as seen when comparing Figure 1 and Figure 2. In the future, I believe that this method could be improved by having a very large stopwords list. This could help make the vectors more distinct

from each other by removing many of the words that are similar between the genres. In a similar note, we could use tf-idf schemes to try and reduce the impact of words the occur frequently across the genres that our stoplist does not catch.

## Conclusion

One of the main things we learned during this project is the importance of a large and diverse dataset. Throughout the project we needed to keep adjusting our data in order to make the distribution more equal and that reduced our dataset to 10% of what it originally was. If we were redoing this project, we would have spent even more time trying to get diverse songs so that our dataset was not so skewed. We also learned just how time-consuming good preprocessing is in order to get usable data. A large portion of time was spent obtaining the songs and organizing them in a way that could make it possible to use their lyrics for the models we built. Due to the fact that most websites did not provide all the information we needed in an accessible way, we were challenged with retrieving all of this information from a few sources. Another thing we could have done differently is to focus on just one method and try to figure out every way possible to optimize it. Due to the fact that we had spent a lot of time obtaining and preprocessing our data, we did not have as much time to experiment as much as we would have liked. If we had more time, we would try and find different ways to optimize results. We would have wanted to dive into tf-idf schemes and see how much of an impact they would have had on our results. One way that we failed was that our vectorization did not produce the results that we hypothesized. However, as it was a unique approach that we had not seen before, we believe the attempt was a success in that we combined a few different NLP techniques together in a way we had not learned about.

Though it was concerning to see that genre analysis through lyrics using our models was not a viable alternative to either human categorization or classification through audio features, we feel that this project did unearth interesting aspects of lyrics as they pertain to musical genre. Being that our basic Naive Bayes implementation only lead to a minor improvement over randomly guessing the genre of a song, it is safe to assume that lyrics alone do not say much about the genre of a song. For instance, changing the cadence or pitch of the delivery of the lyrics can change the genre of a song entirely. Thus, words themselves do not hold too much clout in this classification problem. This is further shown by the marked success our Naive Bayes implementation had when we introduced features such as number of choruses, verses, and words per second. These features help capture the structure of a song, as well as the pace of the song – two aspects that greatly impact song genre – in ways that lyrics alone cannot. Thus, it is not surprising that by adding these features, our model was able to improve as it gained information that word frequencies alone cannot necessarily capture.

## References


M. Fell and C. Sporleder. Lyrics-based Analysis and Classification of Music *Proceedings of the International Conference on Computational Linguistics* pages 620-631, 2014

R. Mayer, R. Neumayer, and A. Rauber. Rhyme and Style Features for Musical Genre Categorization by Song Lyrics *Proceedings of the International Conference on Computational Linguistics* pages 337-342, 2008

Ying, T. C., Doraisamy, S., & Abdullah, L. N.. Lyrics-Based Genre Classification Using Variant tf-idf Weighting Schemes. *Journal of Applied Sciences*, 15(2), 289.


## Tables and Figures

Table 3: Data Distribution

| Genre | Percentage | Total |
|---|---|---|
| Folk | 13.8968% | 10.26 |
| Rap | 8.0455% | 594 |
| Rock | 62.2105% | 4593 |
| R&B | 3.2643% | 241 |
| Country | 5.5262% | 408 |
| Blues | 7.0568% | 408 |

Table 4: Basic Naive Bayes with 1-grams

| Genre | Accuracy | Precision | F1-Score |
|---|---|---|---|
| Folk | 0.67 | 0.01 | 0.01 |
| Rap | 0.99 | 0.98 | 0.61 |
| Rock | 0.67 | 0.00 | 0.00 |
| R&B | 0.67 | 0.00 | 0.00 |
| Country | 0.67 | 0.00 | 0.00 |
| Blues | 0.67 | 0.00 | 0.00 |

Table 5: Advanced Naive Bayes with 1-grams

| Genre | Accuracy | Precision | F1-Score |
|---|---|---|---|
| Folk | 0.74 | 0.21 | 0.13 |
| Rap | 0.81 | 0.44 | 0.28 |
| Rock | 0.67 | 0.00 | 0.00 |
| R&B | 0.67 | 0.01 | 0.01 |
| Country | 0.68 | 0.05 | 0.03 |
| Blues | 0.98 | 0.94 | 0.59 |

Table 6: Basic Naive Bayes with 2-grams

| Genre | Accuracy | Precision | F1-Score |
|---|---|---|---|
| Folk | 0.68 | 0.05 | 0.03 |
| Rap | 0.89 | 0.67 | 0.42 |
| Rock | 0.74 | 0.22 | 0.14 |
| R&B | 0.69 | 0.08 | 0.05 |
| Country | 0.67 | 0.02 | 0.01 |
| Blues | 0.67 | 0.00 | 0.00 |

Table 7: Advanced Naive Bayes with 2-grams

| Genre | Accuracy | Precision | F1-Score |
|---|---|---|---|
| Folk | 0.89 | 0.67 | 0.42 |
| Rap | 0.78 | 0.33 | 0.21 |
| Rock | 0.67 | 0.00 | 0.00 |
| R&B | 0.75 | 0.25 | 0.15 |
| Country | 0.68 | 0.03 | 0.02 |
| Blues | 0.86 | 0.57 | 0.36 |

Table 8: Basic Naive Bayes with 3-grams

| Genre | Accuracy | Precision | F1-Score |
|---|---|---|---|
| Folk | 0.71 | 0.14 | 0.09 |
| Rap | 0.92 | 0.75 | 0.47 |
| Rock | 0.69 | 0.08 | 0.05 |
| R&B | 0.69 | 0.07 | 0.05 |
| Country | 0.7 | 0.11 | 0.07 |
| Blues | 0.67 | 0.02 | 0.01 |

Table 9: Advanced Naive Bayes with 3-grams

| Genre | Accuracy | Precision | F1-Score |
|---|---|---|---|
| Folk | 0.88 | 0.65 | 0.41 |
| Rap | 0.79 | 0.38 | 0.24 |
| Rock | 0.67 | 0.02 | 0.01 |
| R&B | 0.69 | 0.08 | 0.05 |
| Country | 0.72 | 0.15 | 0.09 |
| Blues | 0.85 | 0.56 | 0.35 |
| - | | | |

Table 10: Cosine-Distance Between each Genre and Rock

| Genre | Distance |
|---|---|
| Folk | 0.000939 |
| Rap | 0.004185 |
| Rock | 0.0 |
| R&B | 0.003343 |
| Country | 0.001502 |
| Blues | 0.001723 |

Figure 1: R&B Word Cloud.

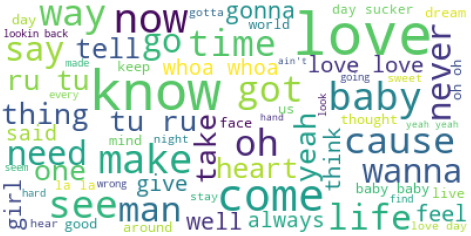


Figure 2: Country Word Cloud.