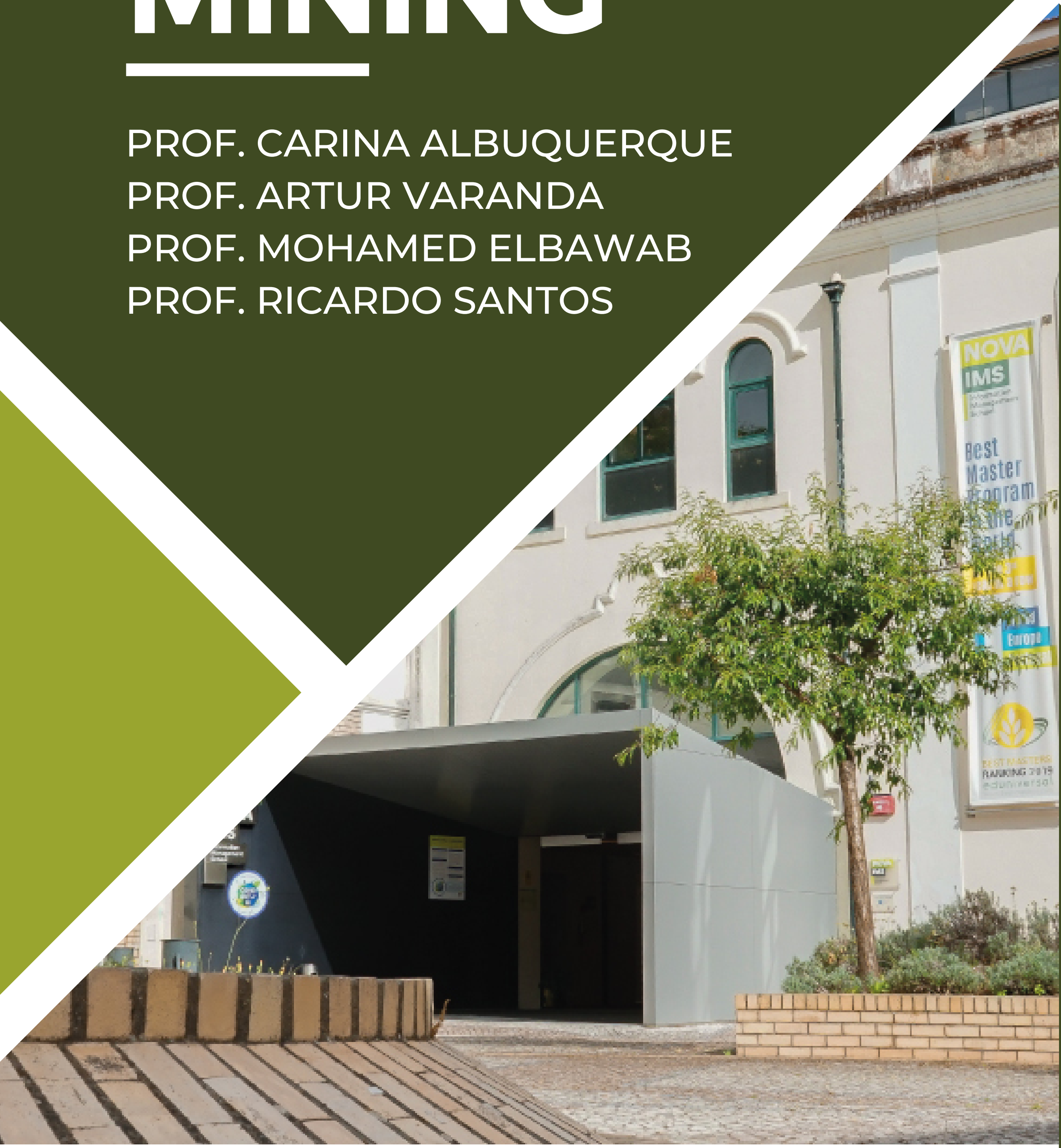
TEXT MINING

PROF. CARINA ALBUQUERQUE
PROF. ARTUR VARANDA
PROF. MOHAMED ELBAWAB
PROF. RICARDO SANTOS



ALEXANDRA PINTO - 20211599
FRANCISCO FARINHA - 20211550
ILONA NACU - 20211602
JOÃO BARRADAS - 20211590
RAFAEL PROENÇA - 20211681

Index

1. Abstract **2**

2. Introduction **3**

3. Data Exploration and Preprocessing **3**

3.1 General Preprocessing 4

3.2 Treating data for Genre Classification 5

3.3 Treating data for Sentiment Analysis 5

3.4 Data Exploration 5

4. Genre Classification **6**

4.1. Trying multiple models 7

4.2. Attempts with Stratified K Fold 7

4.3. Hyperparameter tuning with Grid Search 7

4.4. Final model 8

5. Sentiment Analysis **8**

5.1. VADER vs Textblob 8

5.2. What are the predominant sentiments in the lyrics of the songs of a specific musical genre? 9

5.3. Are there changes in the predominant sentiments of a genre over the years? 9

5.4. Does a song's sentiment impact its popularity within a genre (or even across genres)? 10

5.5. Does the inclusion of featuring relate to any sentiment? Are songs with a more negative tone more likely to include featuring? 11

Figure 4 11

5.6. Lyrics Complexity and Sentiment: Is there a correlation between the complexity of lyrics and the expressed sentiment? 11

6. Conclusion **12**

7. References **12**

8. Annex **13**

1. Abstract

This study delves into the intricate world of song lyrics, employing Natural Language Processing (NLP) techniques for genre identification and sentiment analysis across various musical genres. With a focus on unraveling emotions and themes within song lyrics, the project addresses genre classification and sentiment analysis. Initial data exploration and preprocessing steps involved comprehensive cleaning, language validation, and outlier checks. Specific preprocessing tailored for each objective—genre classification and sentiment analysis—was implemented.

For genre classification, diverse preprocessing techniques, including lemmatization and stopword removal, were explored and tested with multiple models. The final model is based on Stratified K Folds and Softmax Regression. However, the complexity of text data impacted the classification performance, differing from traditional structured data analysis.

In the realm of sentiment analysis, various tools were compared, and predominant sentiments across genres were identified, showing trends in sentiment changes over the years. Moreover, a correlation analysis between lyric complexity and expressed sentiment indicated a weak negative relationship.

2. Introduction

In the world of music, lyrics are like windows into the artist's soul, they express feelings, narratives, and perspectives that are particular to different genres. This project will investigate these poetic landscapes through Natural Language Processing (NLP). Using NLP approaches, the project aims to unravel the emotions and themes contained within song lyrics from a set of musical genres.

The project's primary objectives revolve around two key areas: genre identification and sentiment analysis.

The main goal within the Genre Identification section is to build a classification model that can correctly identify a song's genre from its lyrics. This project requires the use of suitable approaches since we are dealing with a multiclass classification problem.

The Sentiment Analysis part focuses on interpreting the subtle emotional undertones found in songs of various genres. The task involves not only extracting sentiment ratings from song lyrics but also unearthing substantial emotional trends prevalent within individual genres. The aim is to investigate and understand the dominant emotions in each genre and find any recurring themes or distinctive emotional signatures that define them.

To execute this analysis two datasets were provided, train and test. The train dataset has seven columns: *'title'*, *'tag'*, *'artist'*, *'year'*, *'views'*, *'features'*, and *'lyrics'*. The test one has all these columns except for *'tag'*.

We found similar work, related to the sentiment analysis component. There are many papers related to the analysis of text in order to glean emotions, such as sadness, happiness, and anger, among many others. Music is such an important aspect of human lives that there is a lot of research on the subject, less of it related solely to the lyrics, but still some. Some use just the lyrics, like us. [1] In this paper, they attempted to build a classifier that would predict a song's emotion, through the analysis of its lyrics. Other researchers also combined the analysis of the audio, which is a more common approach in the area of Music Emotion Recognition (MER). [2] In this paper, the goal was to analyze the relationship between lyric features and perceived emotions, using Chinese songs.

More related to our genre classification part, we found another project with the objective of predicting the song's genre using the lyrics.[3] These researchers used machine learning models to build a classifier to predict a lyrics' genre. Interestingly, they combined word embeddings with TFIDF, finding that it outperformed simple word embeddings, and our project will use TFIDF as our preferred vectorization method, which to us signals that we went in the right direction.

We believe that having read these articles of projects similar to ours will help us have better insights during the development of our project.

3. Data Exploration and Preprocessing

Before exploring genre classification and sentiment analysis we first need to apply preprocessing techniques to our data. Recognizing the dual objectives of this project, we began by implementing preprocessing steps that catered to both goals. Following this, we distinctly applied specific preprocessing steps tailored for genre classification and sentiment analysis. For example, in the context of sentiment analysis, the importance of retaining capital letters as potential indicators of sentiment was acknowledged. Consequently, separate preprocessing treatments were administered to the data, ensuring that distinct datasets were prepared to facilitate the execution of the two distinct objectives.

3.1 General Preprocessing

We cleaned the dataset in the first stage of our preprocessing to make sure our lyrical data was relevant and clean. We started by determining which characters in the song lyrics weren't needed and eliminating them. We employed a variety of regex patterns to systematically remove special characters, backslashes, brackets, and specific words such as 'verse' and 'chorus', among others (all specific regex patterns are displayed in the preprocessing notebook). The purpose of this careful cleaning procedure was to avoid any possible influence on the analysis that came after, consequently, the results of this action were stored in a new column called '*lyrics_without_regex*'.

Making sure no lyrics were in a language other than English was a crucial step since the techniques we were using are tailored to that and different languages have different structures and requirements. To maintain uniformity and linguistic consistency throughout the dataset, lyrics that were determined to be in another language were removed. As were several rows that were deemed not to be in English by the algorithm used, because they had a strong presence of repeating sounds and beats, which held no meaning and were not focused in any one genre and so could potentially hurt our analysis.

The discovery of two rows with missing values in the 'title' column prompted further investigation. Nevertheless, as the 'lyrics' column was the primary focus of our study and contained no missing values, the influence of these missing titles was considered insignificant. As a result, the dataset kept these two rows with missing title values.

We also checked for duplicates and there weren't any.

We did notice that there were some songs with release dates predating 1600, prompting a need for verification due to their rarity. A decision emerged, to retain all historical data for potential insights or filter out pre-1900 entries for contemporary relevance, balancing historical significance against analytical relevance. We decided to maintain all rows, since we are looking directly at the lyrics and having more observations in our analysis seems more important.

Checking for outliers was also a possibility, with cosine similarity for example, and then checking, per genre, which lyrics were outside of a defined threshold and, hence, were considered outliers. This could have helped in the genre classification model. Unfortunately, we weren't able to complete this part because the resulting similarity matrices were too big, and we did not have the computer resources to keep trying. We thought it best to focus our attention on other parts of the project, after a few attempts at making it work and failing.

3.2 Treating data for Genre Classification

For Genre Classification, we experimented with four distinct preprocessing combinations in 'lyrics_without_regex' column: applying lemmatization only, removing stopwords only, a scenario where neither of these techniques were applied and one they were both applied.

Stopwords are common, low-information words like "the" and "and" removed during text analysis to highlight significant content words, improving the relevance and efficiency of data processing. Lemmatization is a linguistic process that simplifies words to their base or root form, aiding in text analysis by grouping variations of a word for better understanding and interpretation.

Throughout these variations, all words within the *"lyrics_without_regex"* column were uniformly converted to lowercase, and punctuation was removed. These diverse combinations were stored into distinct columns within the dataset, organized to enable comparative testing and analysis within the Genre Classification notebook to see what preprocessing technique worked best.

3.3 Treating data for Sentiment Analysis

In this section, we introduced a new column named 'featuring' within the dataset. This column holds binary values, 0 and 1, signifying whether a song features an additional artist or not, respectively.

Regarding Sentiment Analysis, we made deliberate choices to retain punctuation, capital letters, and stopwords within the '*lyrics_without_regex*' column. These elements were preserved as they might significantly impact the analysis, thus maintaining this column in its original form for the analysis process.

3.4 Data Exploration

We initiated our analysis by examining the distribution of genres within our target variable, labeled as 'tag,' representing the musical genre associated with each song. To visualize this distribution, we created a bar chart showcasing the frequency of different musical genres.

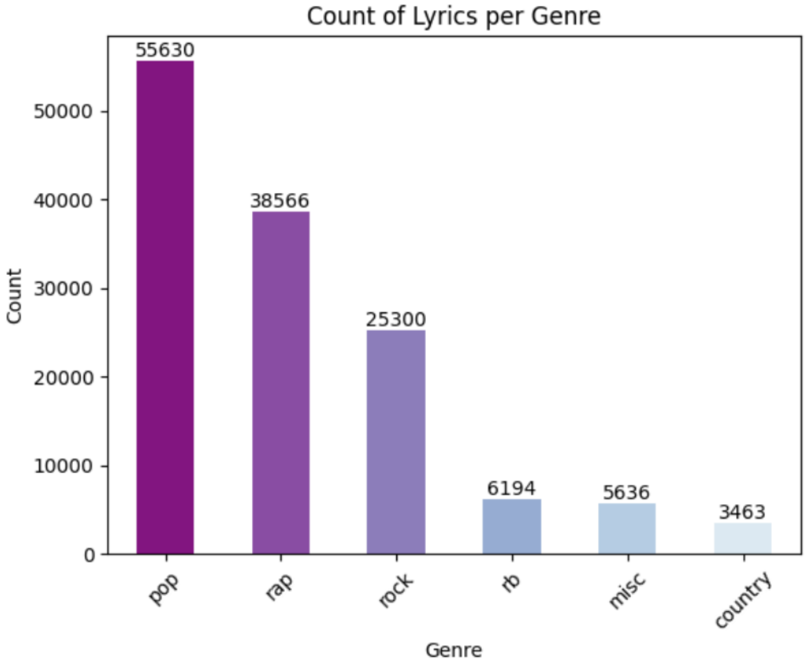


Figure 1

The plot, Figure 1, highlights certain classes, such as '*rb*' (R&B), '*misc*' (Miscellaneous), and '*country*,' which show notably lower representation compared to other genres. This observation indicates a class imbalance within our dataset, a crucial insight to consider for subsequent analysis.

We proceeded with analyzing word clouds for each genre, showcasing the most frequent words found in the lyrics. The primary aim was to check if our genres were sufficiently distinguishable among themselves, by looking at the top words. This step aimed to enhance the distinctiveness of each genre's lyrics, thus aiding Genre Classification. A more distinguishable set of genres ensures better model performance, as shared frequent words across genres might lead to misclassification. Displayed in the annex [Fig.1 to 12] are the word clouds for each genre, built with two methods: Bag of Words (bow) and TF-IDF (tfidf). We focused more on the TD-IDF word clouds, as it seemed more meaningful than just seeing the frequency, but we also looked at both of them, to see how different they were, which also helped cement our certainty that, for this case of lyric analysis, TF-IDF is more useful.

Bag of Words captures word frequency information, ignoring grammar and order. TF-IDF (Term Frequency-Inverse Document Frequency) is a metric assigning weight to words based on their frequency in a document and rarity across a corpus. It highlights important terms for information retrieval, document categorization, and analysis, aiding in meaningful content extraction from vast textual datasets.

It's clear from examining the generated word clouds that certain words frequently show up in different genres. This phenomenon has an impact on a classification model's performance, as previously mentioned. We carefully decided to remove words that were highly frequent in a variety of genres. "*Im*", "*get*", "*know*", "*like*", "*love*", and "*go*" are among the words that have been removed. We only removed them from the dataset that is used for genre classification

Another plot present is a grid of bar charts [Fig.13 to 18], done mostly out of curiosity and a desire for comparison, which shows the top words for each genre, and the number of times they appear, after removing the words mentioned above. It is done to check if the genres appear to be different enough among themselves and see what words are the most present.

4. Genre Classification

In this section, our main goal is to create a model that can classify the genre of a song based on its lyrics. To do that, we built multiple multiclass models considering the multiclass classification problem we were dealing with. Each song can only correspond to one genre and that constitutes a multiclass classification problem. The models were built involving various combinations of preprocessing techniques and hyperparameters, and we tried to find the best combination.

Our main metric for evaluating model performance centered around the weighted average F1 score, primarily due to the imbalanced nature of the data. The micro F1 score aggregates true positives, false positives, and false negatives across all classes to compute a single F1 score. It treats each instance equally. The macro F1 score calculates the score for class separately, and then the average across all classes is computed. Every class has equal weight. Whereas the weighted F1 score is a weighted average of the F1 scores for each class, where each class's score contributes proportionally to its size in the dataset. It's ideal for imbalanced datasets, as it considers class imbalance in the final score, so, naturally, we decided to use F1 weighted.

4.1. Trying multiple models

We started by attempting manual hyperparameter tuning on the models learned in class, One Vs Rest with Logistic Regression and Softmax Regression, also known as Multinomial Regression. One vs Rest involves training multiple binary classifiers, each distinguishing one class from the rest. Softmax regression utilizes a logistic regression model that extends to handle multiple classes by applying a softmax function, providing probabilities for each class, and selecting the class with the highest probability as the predicted output.

After a few initial attempts, using the column with lyrics that had their stopwords removed and had lemmatization, the model that performed best was softmax, getting a weighted F1 score of 0.621 on our validation data (obtained after a train-test split) and 0.6225 on unseen data, in the class Kaggle Competition.

To address class imbalance, we used class weights, as without them, certain classes were not being very well predicted.

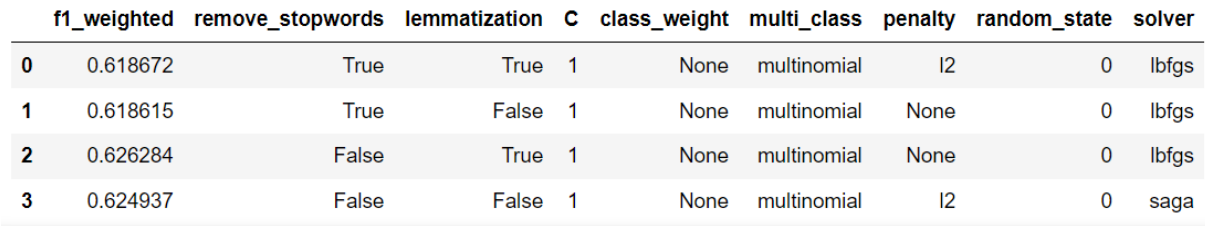
We were curious as to how well our model worked compared with others out there, so we attempted a Multinomial Naive Bayes, which did not perform very well, we only got an F1 score of 0.52, so we did not feel the need to submit the predictions to Kaggle, as it was near to impossible to outperform the previous two models and would not be, for sure, our final model.

4.2. Attempts with Stratified K Fold

Since our data has class imbalance, another way to deal with it, and perhaps a bit better, since it also allows us to get a more robust estimate of how well our model performs, is by using Stratified K Folds, which performs cross-validation dividing the dataset into K folds while preserving the proportion of each class in every fold. We have three attempts in the notebook, each using data with a different preprocessing technique. Two of them use a One vs Rest approach and the last uses a Softmax Regression. Once again, the Softmax Regression performed better. This time, the preprocessed column used was one where the lyrics had not had their stopwords or their lyrics removed.

4.3. Hyperparameter tuning with Grid Search

As a final attempt at improving our results, we did a grid search with the appropriate parameters for Softmax Regression. The results can be seen in Table 1.



	f1_weighted	remove_stopwords	lemmatization	C	class_weight	multi_class	penalty	random_state	solver
0	0.618672	True	True	1	None	multinomial	l2	0	lbfgs
1	0.618615	True	False	1	None	multinomial	None	0	lbfgs
2	0.626284	False	True	1	None	multinomial	None	0	lbfgs
3	0.624937	False	False	1	None	multinomial	l2	0	saga

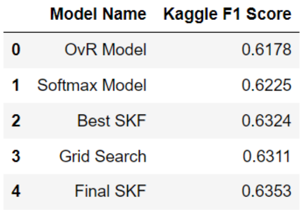
Table 1

We then trained a model with the best resulting parameters from Grid Search, which were keeping *C* and *class_weight* at default values, penalty at None, and using the solver *'lbfgs'*. The best preprocessing technique is deemed to be not removing stopwords and applying lemmatization.

With this model, we obtained a score on Kaggle of 0.6311, which was not our best score. Our best one was obtained using the best model from Stratified K Folds, so we decided to try to apply the suggested parameters and combine them with Stratified K Folds. It worked and we got our best submission, at 0.6353.

4.4. Final model

In Table 2, it's possible to see a comparison of our best models. It is then clear that we picked the model "Final SKF" which stands for Final Stratified K Folds.



	Model Name	Kaggle F1 Score
0	OvR Model	0.6178
1	Softmax Model	0.6225
2	Best SKF	0.6324
3	Grid Search	0.6311
4	Final SKF	0.6353

Table 2

The best F1 score we could obtain was 0.6353, which is much lower than what we are used to in other projects, especially those with structured data, where the minimum was always 0.70 of F1 score. It is interesting to see the differences in performance when the datatypes are so different.

5. Sentiment Analysis

In this sector of the project, we were tasked with analyzing the sentiment of the lyrics of the songs in our dataset. In order to accomplish this, we had to answer three questions:

a. What are the predominant sentiments in the lyrics of the songs of a specific musical genre?

b. Are there changes in the predominant sentiments of a genre over the years?

c. Does a song's sentiment impact its popularity within a genre (or even across genres)?

d. Does the inclusion of featuring relate to any sentiment? Are songs with a more negative tone more likely to include featuring?

e. Is there a correlation between the complexity of lyrics (measured by vocabulary richness) and the expressed sentiment? Do more complex lyrics tend to convey more nuanced sentiments?

5.1. VADER vs Textblob

First, we tested two Natural Language Processing (NLP) tools: VADER and Textblob. NLP tools are commonly used for Sentiment Analysis, which involves determining the sentiment or emotional tone expressed in a text. VADER is a pre-built, lexicon-based sentiment analysis. This NLP tool is known to be good at handling sentiment in short, informal text. Textblob is not much different from VADER, in terms of what it does, but it performs better on more formal texts, such as papers and books. Hence, we did not think it was the best choice for our data, which has a lot of slang and more informal language, but we used it as a benchmark to see how much better VADER was performing.

We took a look at results, through various visualizations such as a table and boxplots, from VADER, per genre, to see the general distribution, in terms of sentiment. Rap and Rock seemed the most negative, and Country the most positive genre, overall. [Fig. 19 to 23] For TextBlob, we took a look at the same visualizations, a table and boxlots [Fig. 24 to 26],, but the results were worse as most genres were identified as neutral, with very little positive, or negative emotions.

For comparison, we looked at the scores, boxplots and wordclouds resulting for both models, for negative sentiments, neutral and positive ones, and, as suspected, the results from VADER seem better and clearer. [Fig. 24 to 31]

5.2. What are the predominant sentiments in the lyrics of the songs of a specific musical genre?

To answer this question we decided to calculate the average compound score of each genre and got these results: *country: 0.413600, misc: 0.296455, pop: 0.263679, rap: -0.206184, rb: 0.389329, rock: -0.009758.*

We felt this was a good indicator of the predominant sentiment in each music genre, but there are some interesting factors, for example, even though the country is the genre with the highest average compound score, R&B as the biggest average positive compound score. This can be explained due to the way VADER works when analyzing the sentiment of a text using VADER, we receive an output with the compound score, the negative score, the positive score, and the neutral score, so maybe rb songs have pics of positiveness in their songs, but these are compensated by more negative parts of the lyrics while country music keeps itself positive rarely featuring negative sentiments, this is proved when you look at the results of the negative compound score for country and rb and see that this value is greater for rb music.

5.3. Are there changes in the predominant sentiments of a genre over the years?

To answer this question we decided to plot line plots showing the change of compound score of each genre during the years. We did a plot with all the genres together [Fig. 32], and we also separated them, in order to be able to compare them and to see their individual evolution.

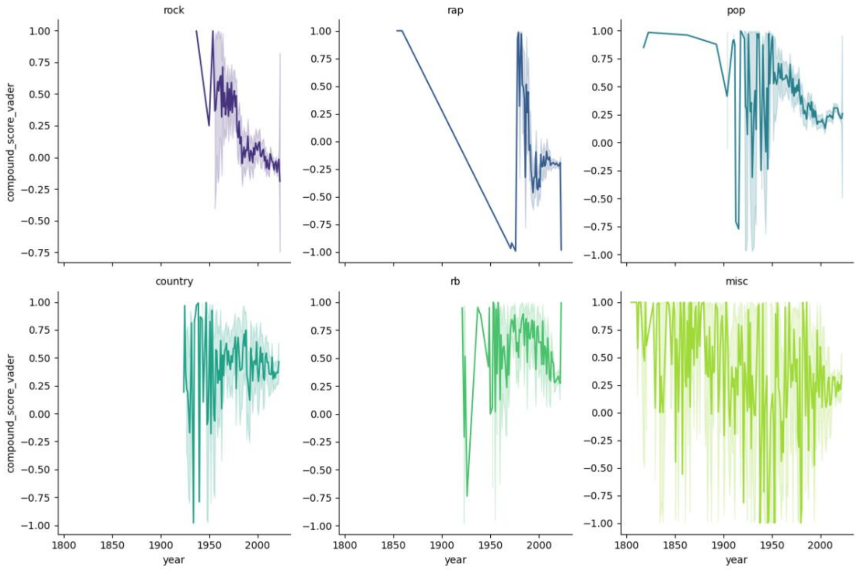


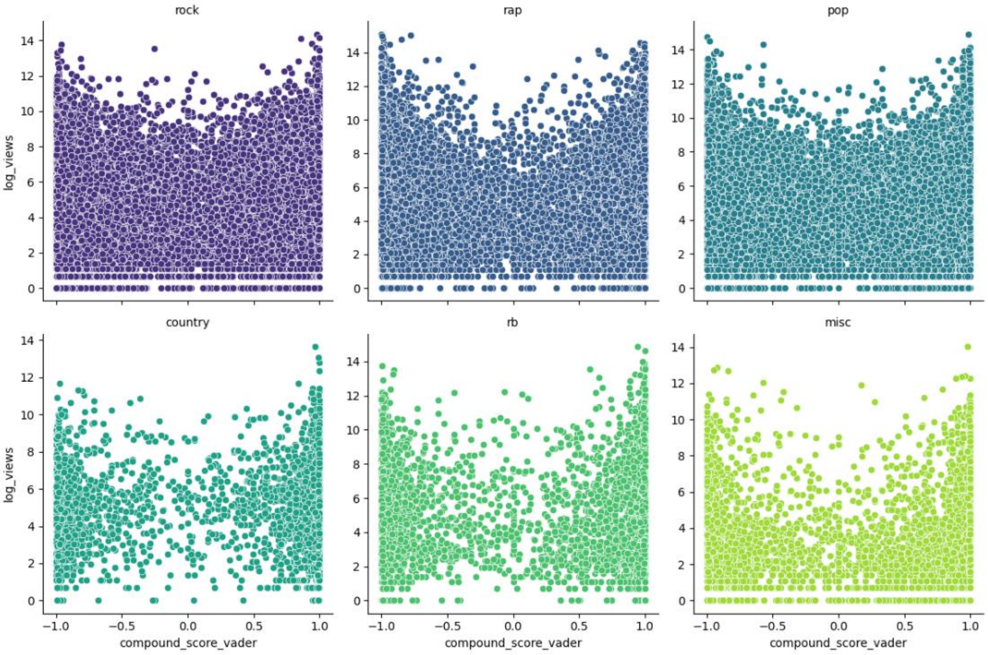
Figure 2

- Looking at the line plots, we see that rock music used to have a pretty positive sentiment, but with time it got more and more negative.
- Rap also started quite positive, until the 90's when the compound score dropped and it remained with a relatively negative sentiment until the actual days.
- Pop songs started quite inconsistent until the 50's. Then they started to get consistently positive, descending a bit in the last decades.
- Country music also started quite inconsistent until the 50's. After that, it kept a consistent positive trend.
- Regarding R&B, this genre kept pretty positive, rarely reaching the zero compound score mark.
- Miscellaneous was the most unpredictable genre during the years, it had very positive songs and very negative songs and never really had more negative and more positive years. Which makes sense considering that it's a mixture of a lot of things.
- Finally, we discovered a trend, almost all genres seemed to begin with an overall positive sentiment and it fell over the years.

5.4. Does a song's sentiment impact its popularity within a genre (or even across genres)?

In order to answer this question we had to decide what was a good way to measure the popularity of a song, for this we chose the number of views that the song had. We did a scatterplot for each genre comparing the views of a song and compound score. Analyzing it, what first caught our attention was that the most positive and most negative songs have the most views, and as the songs get more neutral the view count gets lower, as can be seen in Figure 3. Rock and Pop are the genres where this trend is not so accentuated.

Figure 3



5.5. Does the inclusion of featuring relate to any sentiment? Are songs with a more negative tone more likely to include featuring?

The mean compound sentiment score for songs without featuring is approximately 0.106, while for songs with featuring, it's approximately 0.006, as can be seen in Figure 4. This suggests that, on average, songs without featuring tend to have a higher compound sentiment score (more positive sentiment) compared to songs with featuring, which exhibit a lower mean compound sentiment score (closer to neutral sentiment).

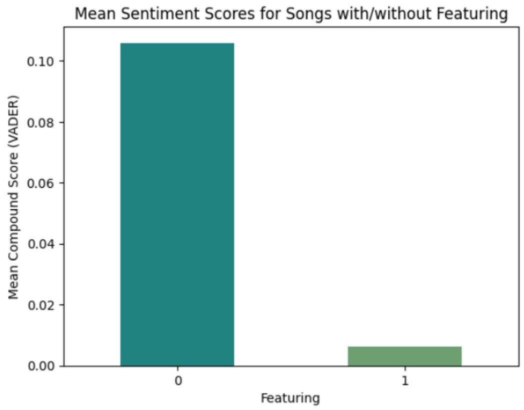


Figure 4

5.6. Lyrics Complexity and Sentiment: Is there a correlation between the complexity of lyrics and the expressed sentiment?

Finally, we decided to analyze whether there was any correlation between the complexity of a song's lyrics and the sentiment of the song itself. In order to do this analysis we calculated the Pearson Correlation coefficient between the vocabulary richness of the song lyrics and its compound score and got the value of approximately -0.106.

So there is a weak negative correlation between vocabulary richness and the compound score. This means that on average as the vocabulary richness increases the compound score decreases slightly. -0.106 is not a strong correlation coefficient, so we can conclude that there is not a strong correlation between vocabulary richness in a song and its compound score.

This was also calculated for each genre, [Fig. 33], and while some genres show slightly stronger negative correlations (e.g., Pop and Rock), overall, the correlations remain weak, suggesting a minimal relationship between the complexity of lyrics and the expressed compound sentiment across these genres.

6. Conclusion

The findings of this study revealed intriguing insights into song lyrics' sentiments and genres, aligning with some initial expectations, such as VADER being better than TextBlob for our data, while revealing unexpected nuances. The limitations in achieving higher classification performance were apparent due to the inherent complexity of text data compared to structured datasets typically used in machine learning.

One notable limitation was the inability to analyze outliers in genre classification due to resource constraints, which might have impacted the model's predictive capacity. Future work could explore alternative approaches to handle text complexity.

The study suggests possibilities for further research, such as experimenting with different models, attempting word embeddings as a vectorization technique, refining preprocessing techniques, or incorporating additional contextual features beyond lyrics to enhance genre classification accuracy.

In conclusion, while achieving high classification performance in text-based genre classification remains a challenge, this study offers valuable insights into sentiment analysis and genre identification in song lyrics.

7. References

1. Revathy, V. R., Pillai, A. S., & Daneshfar, F. (2023). LyEmoBERT: Classification of lyrics' emotion and recommendation using a pre-trained model. Journal Title. Advance online publication. Retrieved January 31, 2023.[Science Direct]

2. Xu L, Sun Z, Wen X, Huang Z, Chao CJ, Xu L. Using machine learning analysis to interpret the relationship between music emotion and lyric features. PeerJ Comput Sci. 2021 Nov 15;7:e785. doi: 10.7717/peerj-cs.785. PMID: 34901433; PMCID: PMC8627224. [NBC]

3. A. Kumar, A. Rajpal and D. Rathore, "Genre Classification using Word Embeddings and Deep Learning," 2018 International Conference on Advances in Computing, Communications and Informatics (ICACCI), Bangalore, India, 2018, pp. 2142-2146, doi: 10.1109/ICACCI.2018.8554816. [IEEE]

8. Annex

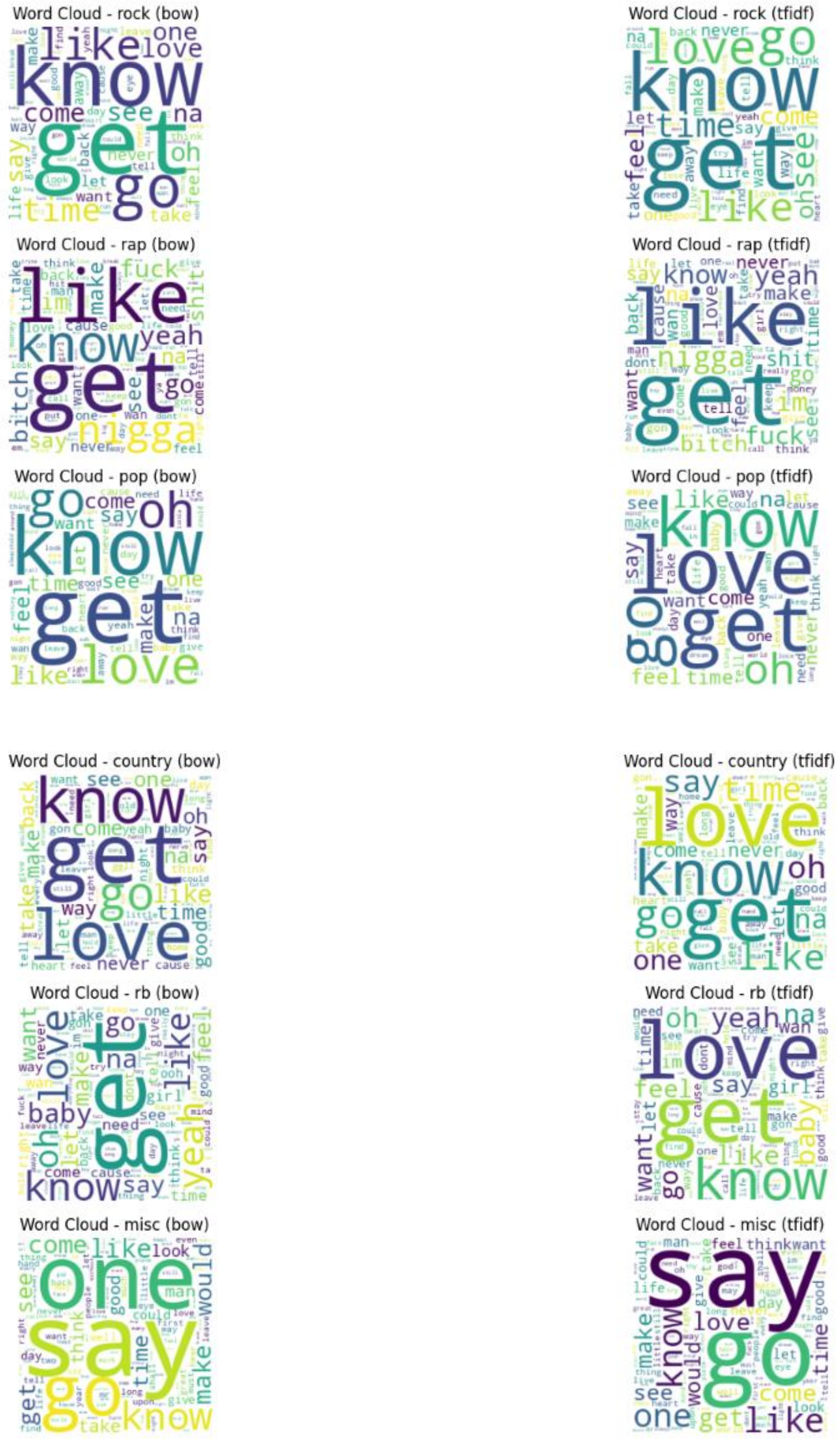


Fig. 1 to 12 - Displayed here are the word clouds for each genre, built with the two methods: Bag of Words (bow) on the left and TF-IDF (tfidf) on the right.

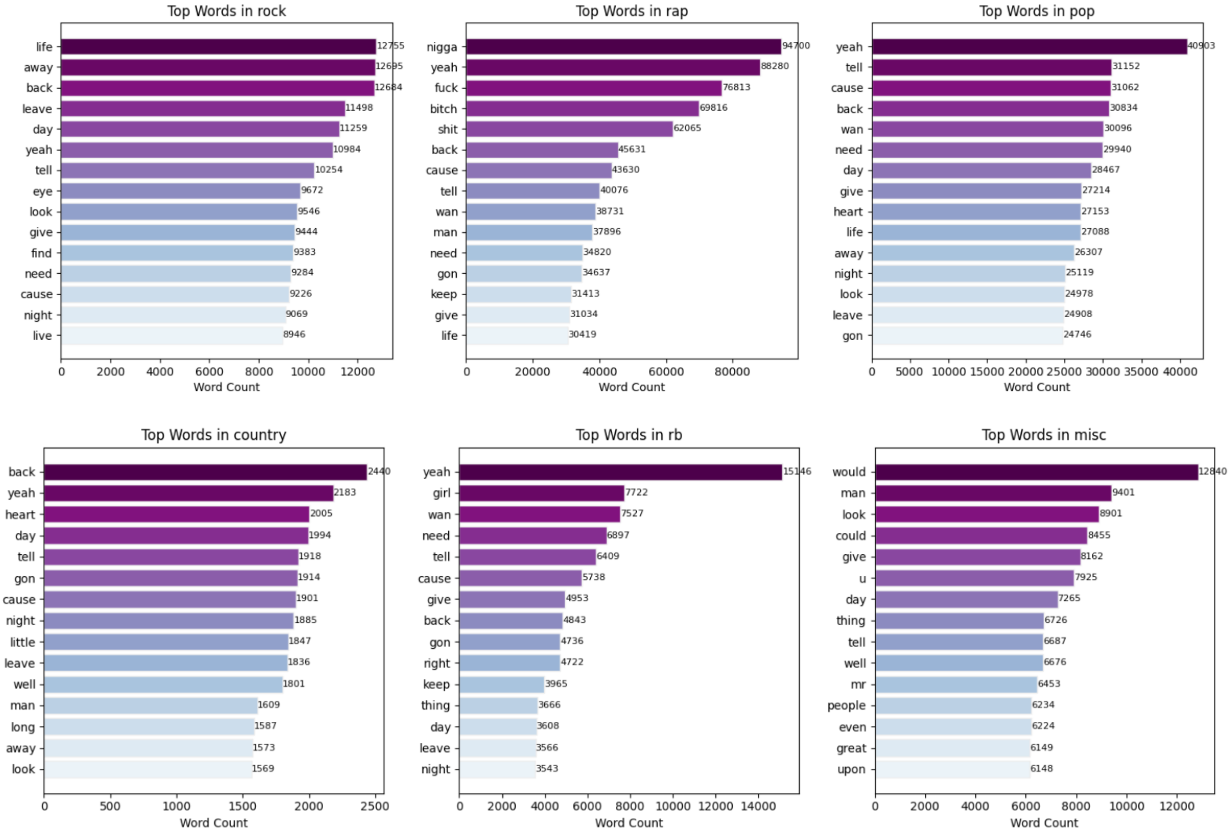
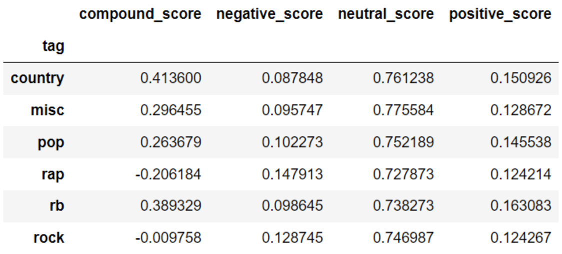


Fig. 13 to 18: Bar Chart with top words in each music genre

Multinomial Naive Bayes

Multinomial Naive Bayes (MNB) is a probabilistic machine learning algorithm commonly used for classification tasks, particularly with text data. It's an extension of the Naive Bayes algorithm, designed to handle features representing counts or frequencies. MNB calculates probabilities based on the frequency of occurrences of features within each class. MNB operates on the assumption of feature independence given the class. Despite this simplification (which might not hold in real-world scenarios), it tends to work. MNB estimates the probability of a sample belonging to a particular class given its features by combining prior probabilities and conditional probabilities of the features given the class using Bayes' theorem. The class with the highest probability is the one assigned as answer.



	compound_score	negative_score	neutral_score	positive_score
tag				
country	0.413600	0.087848	0.761238	0.150926
misc	0.296455	0.095747	0.775584	0.128672
pop	0.263679	0.102273	0.752189	0.145538
rap	-0.206184	0.147913	0.727873	0.124214
rb	0.389329	0.098645	0.738273	0.163083
rock	-0.009758	0.128745	0.746987	0.124267

Fig. 19: VADER Scores per Genre

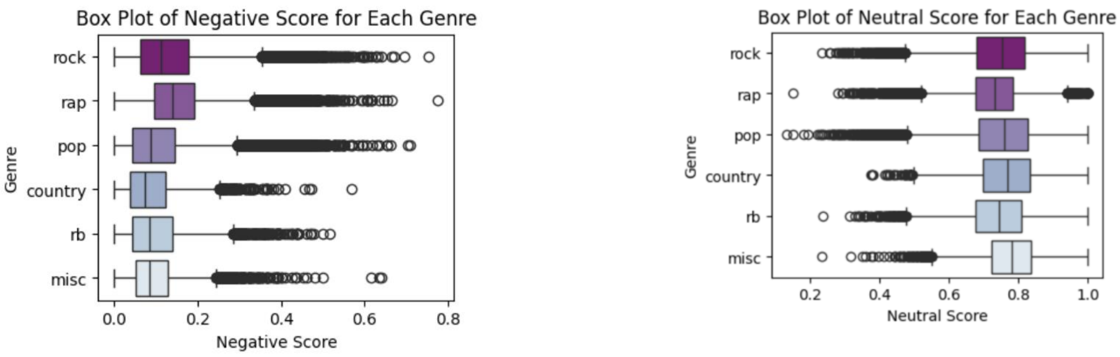


Fig.20 and Fig. 21: Box plot of Negative and Neutral Score for each Genre

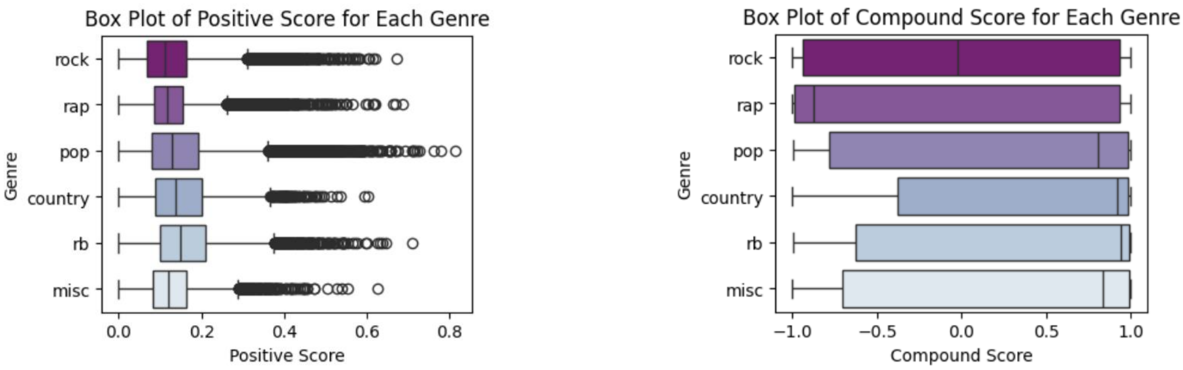
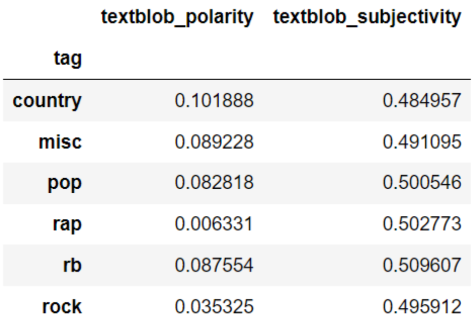


Fig. 22 and Fig. 23: Box plot of Positive and Compound Score for each Genre



	textblob_polarity	textblob_subjectivity
tag		
country	0.101888	0.484957
misc	0.089228	0.491095
pop	0.082818	0.500546
rap	0.006331	0.502773
rb	0.087554	0.509607
rock	0.035325	0.495912

Fig. 24: TextBlob Scores per Genre

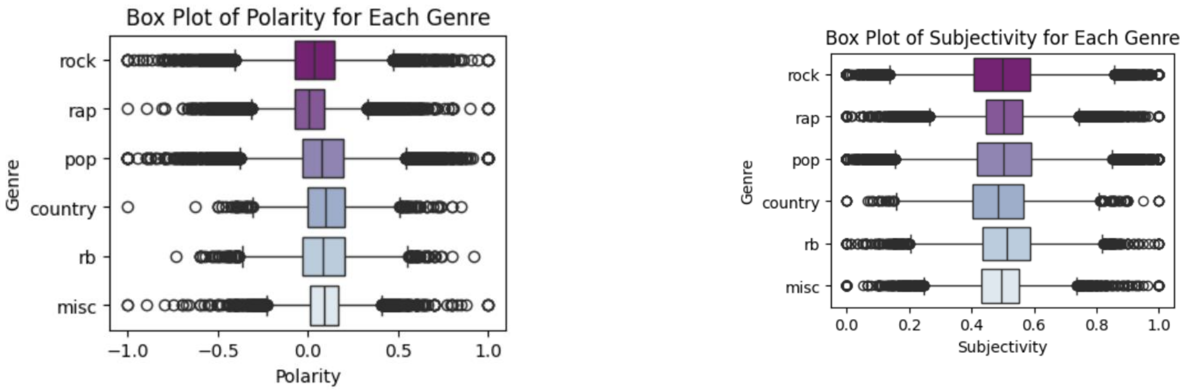


Fig. 25 and 26: TextBlob - Polarity and Subjectivity per Genre

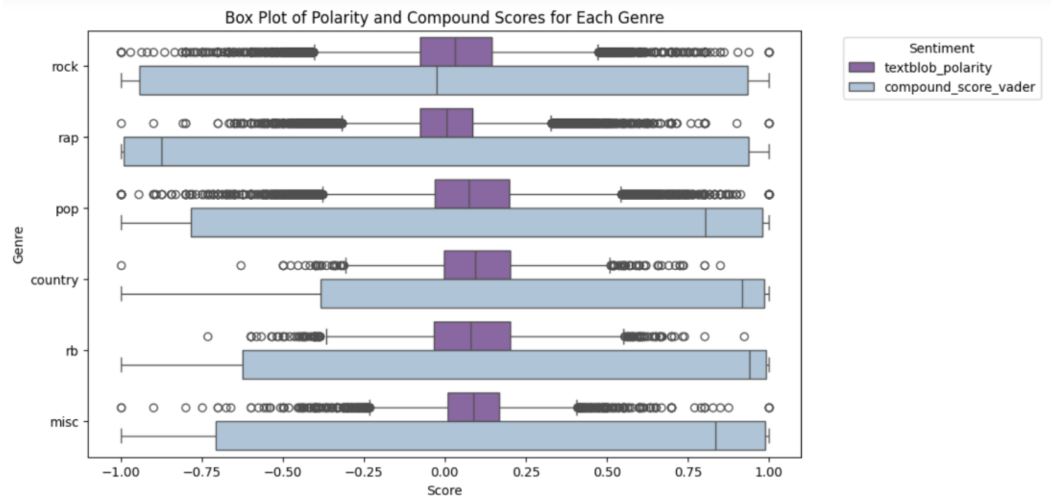


Fig. 27: Boxplot comparison between VADER and TextBlob

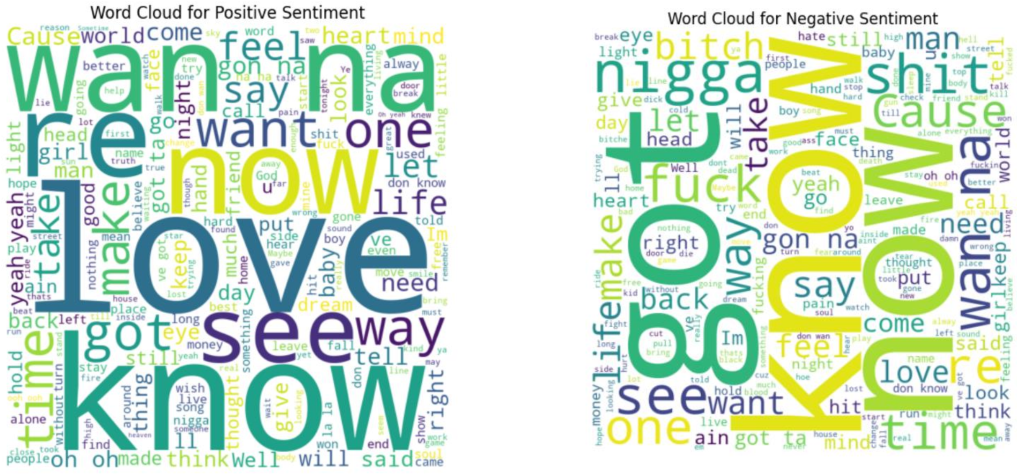


Fig. 28 and Fig 29: Wordclouds for VADER

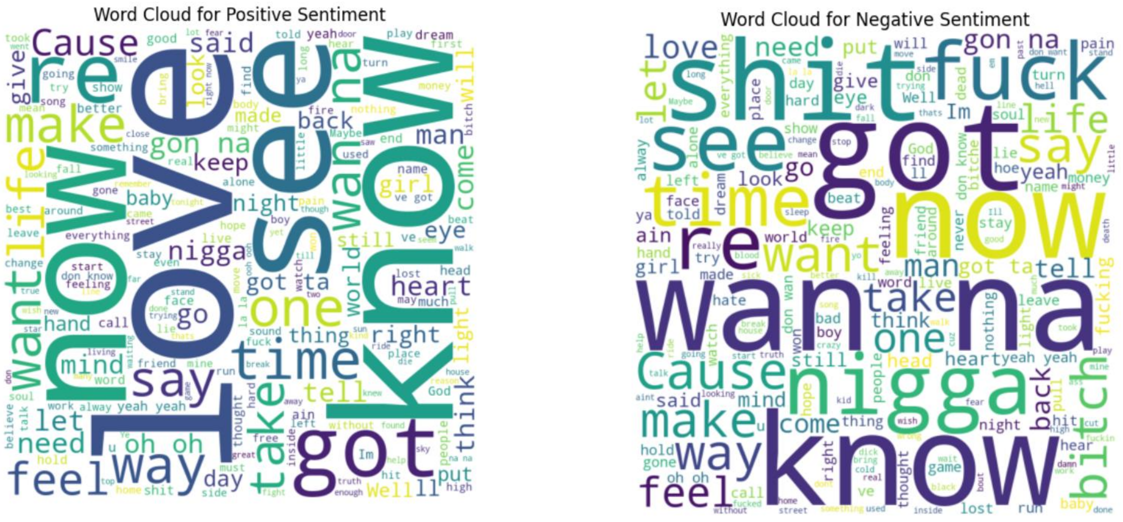


Fig. 30 and Fig. 31 - Wordclouds for TextBlob

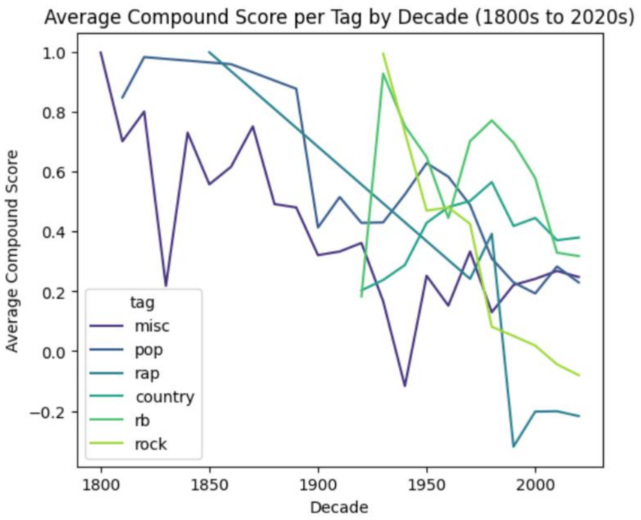
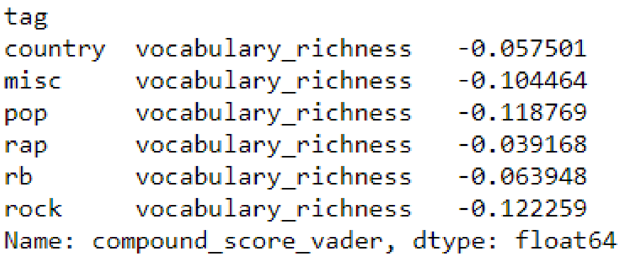


Fig. 32 - Average Compund Score by Genre per Decade



```
tag
country  vocabulary_richness   -0.057501
misc     vocabulary_richness   -0.104464
pop      vocabulary_richness   -0.118769
rap      vocabulary_richness   -0.039168
rb       vocabulary_richness   -0.063948
rock     vocabulary_richness   -0.122259
Name: compound_score_vader, dtype: float64
```

Fig. 33 - Correlation between vocabulary richness and compound sentiment scores