NOVA IMS
Information Management School

# TEXT MINING

## FINAL PROJECT REPORT

Bachelor's Degree in Data Science

Fall Semester 2023/2024

### GROUP PROJECT NO. 3


Catarina Oliveira - 20211616

Daniel Kruk - 20211687

Joana Rosa - 20211516

Marcelo Junior - 20211677

Martim Serra - 20211543

# Table of Contents

## Abstract

Music is growing industry nowadays and the ability of extracting information from song's lyrics might be quite useful to satisfy customers' needs and platforms' organization. Similar tasks are described in the literature and one common approach to deal with genre categorization has been using Neural Networks such as Hierarchical Attention Neural Network. Some papers also analysed sound, however lyrics tended to get better results. As for sentiment analysis, literature tends to be divided in rule-based approaches and the use of machine learning algorithms. The goal of this project was to develop a predictive model to classify the genres of the songs and to analyse the sentiments inherent for each song. After exploring the data and identify insights such as target feature being imbalanced or having songs from ancient history, data was cleaned and treated. At this stage case normalization, contraction handling, random elements removal, tokenization, different language checking, stopword removal and lemmatization with Part-Of-Speech Tagging were performed. For the genre classification a log-ratio analysis was made to get the most relevant words in the dataset. These words were vectorized through TF-IDF and fed into a multinomial logistic regression. As for the sentiment analysis, the approach was to use a rule-based approach and 3 lexicons were tested. Another technique used word embedding with the aid of recurrent neural networks (RNN). The predictive model for genre classification achieved 0.69 and 0.65 scores in training and validation sets, respectively. As for the sentiment analysis lexicons and flair disagreed in about 30% of the dataset, with Vader being the one preferred since it gave the score of polarity and had more variation. Country and r&b seemed to be the most positive genres with rap being less. The methods used although quite simple provided good results and seem to suit well the task at hand.

## Introduction

In today's fast-paced world, the music industry is growing like never before – with around one hundred thousand new tracks daily. Thus, corresponding to more than 3 million songs in a single year. Given the scale of these numbers, the impossibility of human intervention for the meticulous categorisation of this vast musical range - to better organize songs, albums and artists into broader and similar groups [(1)] - is obvious.

Hence, resorting to Natural Language Processing techniques – the use of specific methods and algorithms for text-preprocessing to extract valuable patterns and high-quality information from a corpus of text - is one of the most efficient ways to improve recommendation algorithms for streaming service users, as well as, for music enthusiasts.

In the context under study, there are two main tasks: a predictive and a descriptive one. The predictive model should be able to accurately classify the genre - in a small set of mutually exclusive classes - to which a song belongs. As for the descriptive task, it is based on enhancing the comprehension of the underlying sentiments embedded in the lyrics of the songs.

These goals are very interesting tasks and can have a major impact on the music business, because they can be useful in different aspects of everyday life but mostly for recommendation systems (suggest music that matches a specific listener mood or automated mood playlist creation) [(6)].

Overall, the aim of this report is to summarize the sequence of choices, illustrate the decision process and complexities overcome during the development of the above-described Text Mining project.

By consulting past research on automated song genre classification, it was possible to conclude that Hierarchical Attention Neural Networks and Neural Networks in general tend to have better performances than standard models (such as SVM) on such supervised learning problems. This type of model is also likely to give more attention to a single word in a song line as it will practically ignore all others.

In the context of sentiment analysis, most scientific research refers the use of the SentiWordNet lexicon which returns positive and negative sentiment scores for each word. Even further, it was stated that lyric sentiment analysis remains superior to audio-based sentiment analysis, where it is more difficult to identify emotion and sentiment hints in the audio [(10)]. Some other papers focused on supervised learning problems, mostly predicting emotions instead of sentiments, and using rule-based models as lexicons like ANEW (Affective Norms for English Words) and SentiWordNet combined with machine learning ones like SVM , neural networks and Naïve bayes Classifiers. One problem with machine learning models might be considered the bias introduced in some words, for example classifying the word "African" with a bad polarity. [(14)]

Additionally, a common issue found among almost all papers that were read was that streaming services tend to categorize the genres through their artists. However, the same artist might have songs of different genres. Moreover, it was recognized that current algorithms being applied in the market practically only use audio data due to its low dimensionality when compared to the large amount of lyrics data. Nevertheless, these have been proven to correctly classify around 61% of the results [6].

Finally, previous studies show suggestions to add different types of song data (audio features or images, such as album art covers) to improve model performance. However, there is also proof that text-based approaches are more appropriate for song genre classification.

## Data Exploration and Preprocessing

- Description of the Received Data

Like in any project whose main goal is to retrieve information from data, the first steps revolve around looking at the initial aspect of the dataset itself and its original characteristics. In order to explore and understand the original data, the dataset was imported, and several visualizations were created to reach conclusions in a more intuitive manner. From the outputs obtained during this stage, it was possible to see that the dataset had several problems that needed to be addressed. The most significant aspect that was noticed was that the dataset was severely unbalanced since 3 out of the 6 classes made up to around 90% of the data (those classes were Pop, Rap and Rock), as visible in Figure 1. Furthermore, the dataset included a lot of songs that, in theory, were released in the early years of the history of humanity (see Figures 2 and 3) and, for those, the number of views peaked in the 18$^{th}$ century. (Figure 4). These two aspects were deemed as not too meaningful since the numerical variables were unlikely to be useful for the future stages of this project.

From plotting the most frequent words prior to treating the data, that are presented in Figure 5, it was realised that they (like almost any data that is retrieved) would have to be treated and cleaned before becoming usable for the latter and most important stages of this project. There were a lot of noise elements within the lyrics, as well as some of the most frequent terms not even being words (elements like "'s", "n't"; words that would later be considered irrelevant (also known as stopwords) like "in", "my", "that"; among many others).

- Cleaning and Treating the Data

This stage of the project was developed in order to what was most common among other projects that deal with text data. Since for the genre classification and sentiment analysis stages only two of the dataset's columns ("*tag*" and *"lyrics"*) would be used, the preprocessing - as it is usually called - was focused on dealing with the irregularities within these two features. Because the *"tag"* column possessed no inconsistencies (other than the unbalancedness of the dataset) the steps that were addressed in this stage were exclusively applied to the *"lyrics"* column.

The steps taken to correct the errors within the *"lyrics"* column were case normalization, contraction handling, random elements removal, tokenization, different language checking, stopword removal and lemmatization with Part-Of-Speech (POS) Tagging.

*Text Normalization*

In this part of the data cleaning, since naturally there were a lot of letters that were not in lowercase, all of the letters of every word or term of the dataset were transformed. This facilitated future analysis, as well as the models' comparisons between words.

*Contraction Handling*

In this stage, the verified contracted terms (whether it was because they were slang or just because they tried to mimic the singer's pronunciation) it was decided to extend all these contractions to how they would be written if they were written to their full extent. To do so, a dictionary that was found while researching for the project was used and the contracted terms were replaced using simple Regex *.replace* method.

*Random Elements Removal*

For this section, a function - stored in the *functions.py* file - which was responsible for removing the various elements that were present in the original lyrics and that can usually be considered as noise data for projects that deal with text was created. This function essentially cleaned and filtered the input text by removing and replacing various elements including emojis, special characters and isolated consonants.

*Tokenization*

In the tokenization stage, the words of the given lyrics were divided into simpler units for analysis – tokens. This facilitated the analysis of a word or a group of them while also ensuring they are usable for model creation. Again, this step was made using the *tokenize_text* function that can be seen in the *functions.py* document.

*Different Language Checking*

In this step it was attempted to make sure that every lyric (and consequently every word) of the dataset was in the same language - English. This was done because since the models were to be developed for the English language, words of other languages would disturb the performance of the final models. During this verification there was a total of nearly 300 rows that had tokens that were not in the English language. From this point on there were two approaches that could be taken: translation of all terms into English or deletion of these rows. In order to support the decision, it was checked if all these rows belonged to the same class, or more specifically to any of the minority classes because since these were already too small, there was not a wish to reduce these observations even further. Luckily, the majority of these observations belonged to the three most represented classes of the dataset, as seen on Figure 6. Hence, deleting these rows seemed like the obvious choice; therefore, avoiding the risk of translating untranslatable words while also reducing (even if it was for very small margins) the size of the majority classes.

*Stopword Removal*

Here, every word from the dataset that was considered a stopword (at least considering the nltk list) was removed. This was done in order to simplify and reduce the amount of noise that was in the *"lyrics"* column which facilitated the model's analysis of these terms. In the context of the problem at hand - song genre classification - it did not make sense to keep articles, prepositions, or conjunctions as part of the data since they would be adding unnecessary complexity to the model.

*Lemmatization (including POS Tagging)*

To finalize the treatment of the data, the surviving tokens were lemmatized - process which consists of re-shaping of the words into their lemmas (the base form that would be found in a dictionary). To do this, the *lemmatizator* function, which did a POS Tagging to the words, was applied. This would correct the words properly (reducing the probability of correcting similar words into the same lemma, despite them being verbs, adverbs or nouns). Lemmatization was applied instead of stemming to avoid the loss of information in case some words were improperly identified during the POS Tagging (i.e. the word "better" has the word "good" as its lemma which if it were to be stemmed would be converted into "bet").

After all of these preprocessing steps were concluded, the data was then explored for verification purposes and then exported for the following stages of the project.

## Genre Classification

- Log-Ratio Analysis

When faced with the genre classification problem the most efficient plan that was found and decided among the developers of the project was to divide it into two main steps: a log-ratio analysis and then the modelling phase where several different models would be experimented.

The main goal behind the log-ratio analysis was to find the most relevant words in the available lyrics – what was then referred to as *vocabulary*, in order to later feed them into the vectorizer and later obtain the best performance possible with the chosen model.

As so, at the very beginning of this stage of the project, the objective was to prepare the data for the log-ratio analysis. That was achieved by first selecting the 3000 most common words of the dataset. Through this, it was possible to significantly reduce the original number of words in the pre-processed dataset. At the same time, another variable was created containing the frequency corresponding to the number of times these 3000 words appeared in the dataset. Then, they were grouped by genre and it was possible to conclude that the category which had the highest number of relevant words was *pop*; this came as no surprise due to the fact that *pop* is the predominant genre (has the most rows) of the data at hand.

After having done this selection of words several times, and similarly to what was done to calculate the frequency of each word in the dataset, their log ratios were computed (the higher the value, the higher is the representativeness of the word in their respective tag) in the following two phases: in the first one, the frequency of the different words in the whole pre-processed data for each of the tags was calculated. The next step consisted of the following division:

$$\log\left(\frac{\left(\frac{Frequency\ of\ x\ Word\ in\ a\ Specific\ Tag}{Total\ Number\ of\ Words\ in\ that\ Tag}\right)}{\left(\frac{Frequency\ of\ x\ Word\ in\ Other\ Tags}{Total\ Number\ of\ Words\ in\ the\ Other\ Tags}\right)}\right)$$

This is a better approach than just choosing *N* words with the highest value of frequency since it allows a wiser choice of relevant vocabulary – the higher the value of the log ratio the better, because it means that that word is more characteristic of a specific tag than other. What is being tried to be avoided is selecting words that appear with similar importance in distinct tags, because that will not allow the model to distinguish the categories from each other (they will have low representativity). Additionally, as the values are affected by a logarithm, they also become smaller and easier to interpret.

Afterwards, a dictionary with the percentage that each genre represents in the dataset was created and those weights were used to ensure the proportion of words for each category matched the distribution of the genres in the overall data – reducing in this case the words to 2998. Finally, out of the list of sorted words by their log-ratio value (organized from highest value to the lowest one) a list containing 2745 distinct best words was obtained. This is due to the fact that in the original 3000 words, some of them appeared more than once, due to being important in more than one genre. Thus, the final vocabulary was a combination of 6 others, which had slightly different preprocessing changes, and it was exported with 4495 distinct words (around 0.02% of the original data) that will allow the prediction of the song genre through its lyrics.

To facilitate the understanding of what has just been mentioned above Table 1 was developed with the 10 most important words out of the last 3000 selected words. Furthermore, after the distribution of the weights for each category (pop – 41%, rap – 29%, rock – 19%, R&B – 4.6%, Miscellaneous - 4.2%, Country - 2.6% ), each of the genres: *pop, rap, rock, R&B, miscellaneous* and *country* was assigned words 1239, 857, 563, 137, 125,

77, respectively. In the end, some interesting words appeared as the best words of each genre; some examples for each of the genres can be seen in Table 2.

- Modelling

The modelling stage of this multiclass classification problem was initiated with a TF-IDF algorithm, which was responsible for the vectorization of the words of the selected vocabulary (explained in the previous section) by attributing different weights to the words – the most frequent words will be assigned smaller weights, since it assumes they convey a less important message. The defined parameters for TF-IDF were: the chosen vocabulary in the previous step, English stopwords, n-gram range between 1 and 2 (extraction of uni and bigrams) and it ignored words with frequency lower than 5 and sublinear scaling was activated. It should be noted that for this section of the project, count vectorizer was also tried. However, given that it does not consider the importance of individual terms common and stop words would dominate the representation. Hence, it was disregarded as TF-IDF, in particular cases, did increase the performance up until 5% in the validation set.

Furthermore, to carry out further testing, the data was standardized (using standard scaler) and the K-best algorithm was implemented (it defines a qui-square score to see the best *k* features). In some particular cases, PCA was also tried as a way to reduce the dimensionality of our features and as a result it also reduced the overfit of some models. In the end, none of these techniques was applied in the vocabulary that was fed into the final model, since in the case of the final model (performance below 70% of f1-score), the trade-off between loss of interpretability of the results and the reduced overfit was not worth it.

Then, the train dataset was divided using *train_test_split* to create a validation dataset where the model would be tested. Since the team did not have access to the test set and even if so, this is able to minimize data leakage into the model.

Throughout the modelling phase, several different algorithms were tested in order to verify which one would work best for the pre-processing that had been done as well as for the selected 3000 words with the log-ratio technique. Among them, some were: LinearSVC, Multinomial Naïve Bayes, Logistic Regression, SVC (Support Vector Classifier), XCG Classifier, Decision Trees and Random Forests. From these, only the Linear SVC and the Logistic Regressions showed decent performance. Therefore, not showing much overfit, having f1-score results about 5 - 10% higher than the others and almost always being able to classify all of the genres. In this process, of trial and error, with the several possible models, a Grid Search was also implemented in order to try and find the best possible parameters for the Logistic Regression model (given that this model had the most promising results out of all the tried ones).

The models were fitted to the training data and the predictions were made on both the training and the validation one. This was so that it was possible to verify the differences in performance in unseen data by the model. Finally, the performance of the models was assessed with the standard evaluation criteria – f1-score ($f$)[10], due to the dataset being imbalanced (distribution of the classes was not equal).

In the end, the model that was chosen by the group as being the one which was able to capture the most patterns in the data and that could better classify the genre of unknown songs was a Logistic Regression. Its selected parameters were random state equal 42 (guarantees that the Logistic Regressions all have the same characteristics) and *c* equal to 100 (this *c* value was obtained on Grid Search). Such a model had an weighted average f1-score 69% in the training set and 65% in the validation data; showing about 4% of overfit, that can be justified by the fact that the vocabulary came exclusively from the training set and it is normal that it will predict better these familiar words to it than some of the never seen ones on the validation set. In relation to the classification distribution of the classes, it was as follows: Country – 26%; Miscellaneous - 59%; Pop - 69%; Rap - 86%; R&B - 20% and Rock – 40%. It is worth noting that it is normal that these distributions are not

exactly similar because of the dataset's imbalanced structure (classes with more data, tend to be more representative in the predictions).

## Sentiment Analysis

- Methodology

To analyze the sentiments inherent to each music, the methodology taken was to use a Rule Based approach. Since this task was unsupervised it limited the machine learning techniques available and so, Rule Based systems seemed to be a simple but effective approach. To get a better understanding of the sentiment underlying each song 3 lexicons were applied. Lexicons are essentially dictionaries in which each word is related to a sentiment score. Two of the lexicons used that are quite common are VADER (Valence Aware Dictionary and sEntiment Reasoner) and TextBlob. Vader was tunned for social media and since a good number of lyrics contained slang it could provide more accurate results. As TextBlob it provides better results on more formal tasks but, it could be interesting to compare. Both lexicons provide a polarity between -1 and 1, the first representing a quite negative polarity and the other a super positive polarity. When 0, the polarity is neutral. Another Lexicon used was the NRC Emotion Lexicon, which is essentially a collection of dictionaries for each emotion/sentiment with English words associated, and their score. Since this lexicon also presents the emotions, it was found interesting to check not only the sentiment but also the emotion analysis of the lyrics. Other approach was also studied, namely the Flair library. Flair is a NLP library that works with word embeddings and can perform multiple tasks, including sentiment analysis. A deeper understanding of this technique is presented in the annex. The output of this analysis is the polarity and the confidence that the model has on the prediction in a range between 0 and 1 for each song lyrics. [8]

- Results

First of all, it's important to notice that for this task, the data used was the previously pre-processed one. Even considering that in most sentiment analysis cases the punctuation, emojis and capital letters may have impact on the sentiment, in the cases of music lyrics both are not appliable since songs don't usually have punctuation besides commas nor capital letters or emojis that represent any sentiment.

After computing the results from Vader, TextBlob, these were scalled to the same range with MinMax scaler, to be able to compare the results between the two. As for the NRC since the implementation only resulted in positive or negative polarity, without the score, the comparison made was made with Flair, that provided similar results and to check if the results provided by Vader, TextBlob, after scaling, were above or under 0.5. One initial comparison made was between the median, max and min of Vader and TextBlob. Vader had a much greater median, meaning it classified songs more positively than TextBlob, in which the median was closer to 0 (neutral). However, when looking tag wise, the order according to the polarity was almost the same according to both lexicons. By performing some confusion matrices, it was possible do see that the different methods differ on the classifications on around 40 000 songs, which represented around 30% of the dataset. The most positive (considering most positive is to have a higher average polarity) tags in descending order were country, r&b, misc and pop with these last two interchanging in TextBlob score. Rock was more neutral in both and lastly rap was the one with lower score of polarity. That said, when looking through TextBlob all of the scores were more or less around 0. Other interesting insight was to look at the polarity of genres throughout the years This polarity was the one computed by Vader, since as seen before, TextBlob was too much neutral therefore, from now on, Vader was the score chosen for the coming analysis, since it seemed to be the more accurate one with greater variation. This differences among distributions were also possible to look at in the boxplots presented on Figures 7 and 8.

Because most of the songs were after 1900, the graphs only included that era. These are available in the annex, figure 9. These graphics allow some conclusions: throughout the years, rock has been decreasing the

polarity (being closer to neutral), except for the years after 2020 in which it increased drastically. As for rap, the polarity has also been dropping since 1980 but stabilizing in the last two decades, around a negative close to neutral value. In pop, the polarity was quite unstable until 1940, and since then it has been dropping from super positive to more positive neutral (around 0.25). The same instability happened in country music, until around 1960 where it started to stabilize in a quite positive polarity (around 0.5). The last two tags, r&b and miscellaneous (misc), had both quite unstable polarities through time. However, r&b only varied in between positive polarities, meaning it was never negative during this era, as for misc going from extreme-to-extreme polarities.

One other curiosity was to check whether a genre had more views according to the polarity. In fact, some genres showed some interesting patterns. As for country and r&b, a few number of songs achieved a great number of views while having polarities close to 1; nevertheless it's important to not forget that most of the musics in these tags had quite positive polarities and so one of them having many views wouldn't be strange. As for pop and rap, these were quite surprising: songs with the most amount of views were mainly in the extremes of the spectrum, meaning that songs very positive or very negative might end up being more popular. Another surprise was for rock. This was considered a more neutral tag in both lexicons, however the songs with the biggest number of views were located in the right extreme of the graph, meaning they had a quite positive score. These results are available in Figure 10.

One insight that was intriguing was to see the artist with most views in each genre and what both emotion (mode) and average polarity corresponded. Taylor Swift was the most popular artist in two tags: country and pop. In both these tags the polarity was quite high (around 0.72) and the most prevalent emotion was joy, which in fact was the most prevalent emotion for all the top artists in each tag. Rap and R&B also shared the same artist: Drake. But, differently than Taylor Swift, this artist had a super high polarity (around 0.94) on r&b and a still positive but quite close to neutral polarity in rap, meaning his songs tend to be more positive when he sings r&b than when he performs rap. As for the other categories, the most popular rock artist (in this case band) was Led Zeppelin and the polarity was quite high (0.9). Last, but not least, Genius, which is in fact the "world's biggest collection of song lyrics and musical knowledge", was the top artist for the misc category and the polarity was also quite high (0.92). In the same thought, this polarity was also studied for each one of these artists through time. It was remarkable to see that for example Taylor Swift had some moments in her career where the polarity dropped severely, but these moments happened asynchronously in country and pop tags, happening in 2010 and 2020, respectively. Also in this same artist, it's possible to see that she started her career singing country in the decade of 00 and only around 2015 started to sing more pop (according to the songs presented in the dataset, obviously). Even if she had a sop in polarity in pop, it wasn't nearly as low as in country tag. Another artist where this kind of insights were possible was Drake. This artist had clearly performed more time in rap, with the polarity being quite unstable, and achieving is negative extreme in around 2015, which corresponds with an epoch of great success and recognition for the artist. This drop also happened around the same time in r&b tag, even though the path of this artist on this tag is quite shorter than on rap and having a quite more stable and positive polarity. Finally, for Led Zeppelin, they also had a drop in polarity around 1995, but in general maintained a quite positive polarity in their lyrics, as did genius in the misc tag, being much more recent than the previous, as expected. (Figure 11)

Last visualizations on sentiment analysis were word clouds. One word cloud was made for each emotion and Vader score (negative < 0.5 and positive >0.5) – see figures 12 and 13. As mentioned before, emotion analysis was also performed. For the emotion fear, some of the most popular words were "fall", "kill" and "pain" which were quite fearful words, meaning this lexicon was considering good words to represent. Others less obvious words were for example, "love" in anger, or "leave" in joy, which are explainable since they're probably quite common words in the entire dataset, and these categories are not exception. However, overall, the words

presented in the different word clouds resembled the emotions intended. These wordClouds are presented in Figure 14.

For future work, it would be interesting to compare these multiple methods in a supervised task so to be more confident on the obtained results.

## Conclusion

In conclusion, in the long run, this sort of advanced computational techniques will lead to the enhancement of recommendation systems and overall user experience within the spectrum of music consumption.

For the future, and as it has already been proven by several research, combining different types of data on music (song lyrics, audio, images – album covers) might be beneficial to improve the accuracy of the classification models both being and to be developed in this area of study. Additionally, topics for further study can include "music recommendation, audio scene classification, machine listening and cover song classification" [(1).]

## References


(1) Peoples, G. (2023, February 7). Billboard. Billboard. https://www.billboard.com/pro/how-much-music-added-spotify-streaming-services-daily/

(2) Cooke, C. (n.d.). Music Data Explained | CMU Library. https://cmulibrary.com/cmudiy-musicdataexplained/#:~:text=So%20track%20title%2C%20version%2C%20artist,and%20recorded%20any%20one%20recording.

(3) Oramas, S., Barbieri, F., Nieto Caballero, O., & Serra, X. (2018). Multimodal deep learning for music genre classification. *Transactions of the International Society for Music Information Retrieval*. 2018; 1 (1): 4-21.

(4) Boonyanit, & Dahl. (n.d.). Music Genre Classification using Song Lyrics. In Web Standford University. Retrieved November 21, 2023, from https://web.stanford.edu/class/archive/cs/cs224n/cs224n.1214/reports/final_reports/report003.pdf

(5) Tsaptsinos, A. (2017). Lyrics-based music genre classification using a hierarchical attention network. *arXiv preprint arXiv*:1707.04678.

(6) Sharma, V., Agarwal, A., Dhir, R., & Sikka, G. (2016, March). Sentiments mining and classification of music lyrics using SentiWordNet. In *2016 Symposium on Colossal Data Analysis and Networking* (CDAN) (pp. 1-6). IEEE.

(7) Jamdar, A., Abraham, J., Khanna, K., & Dubey, R. (2015). Emotion analysis of songs based on lyrical and audio features. *arXiv preprint arXiv*:1506.05012.

(8) Alan Akbik, Duncan Blythe, and Roland Vollgraf. 2018. Contextual String Embeddings for Sequence Labeling. In *Proceedings of the 27th International Conference on Computational Linguistics*, pages 1638–1649, Santa Fe, New Mexico, USA. Association for Computational Linguistic

(9) Tagging sentiment | flair. (n.d.). Retrieved from https://flairnlp.github.io/docs/tutorial-basics/tagging-sentiment

(10) Xia, Y., Wang, L., & Wong, K. F. (2008). Sentiment vector space model for lyric-based song sentiment classification. *International Journal of Computer Processing Of Languages*, *21*(04), 309-330.

(11) Noble, W. (2006). What is a support vector machine?. *Computational Biology*. 24(12), 1565–1567. doi:10.1038/nbt1206-1565

(12) Pedregosa.(2011) Scikit-learn: Machine Learning in Python*. JMLR 12*, pp. 2825-2830

(13) Rosa, J., Oliveira, C., Júnior, M. & Serra, M. (2022, December 23). *Hotel California - Cancelled Booking Prediction Report*.

(14) Habibi, M. (2023). Sentiment analysis of top-5 charting songs in the US and the UK from March 2020-2022. *[Master Thesis].* Available in*: https://dspace.ut.ee/server/api/core/bitstreams/0558ac34-ab91-412d-bc7a-7e0252669903/content*

(15)*1.10. Decision Trees*. (n.d.). Scikit-learn. https://scikit-learn.org/stable/modules/tree.html

(16)*What is Random Forest? | IBM*. (n.d.). https://www.ibm.com/topics/random-forest

# Annex

*Table 1 – 10 out of the 3000 Most Common Words at Different Steps of the Log-Ratio analysis*

| 3000 Most Common Selection | Number of Times the Word Appears |
|---|---|
| *get* | *417062* |
| *like* | *286100* |
| *know* | *264743* |
| *go* | *221786* |
| *love* | *180760* |
| *say* | *165295* |
| *yeah* | *163217* |
| *make* | *153830* |
| *na* | *153365* |
| *see* | *140957* |

*Table 2 – 5 Final Best words for each of the genres of the dataset*

| Pop | Rap | Rock | R&B | Miscellaneous | Country |
|---|---|---|---|---|---|
| chee | slatt | gah | tingalinga | nbsp | waterbound |
| oya | baow | nothingness | lovebite | munny | aikendrum |
| sleigh | tec | damnation | yooby | alvina | eedle |
| woh | patek | desolation | eeyeah | chichikov | gloryland |
| bruk | opps | blacken | bonz | mcclane | cruzar |

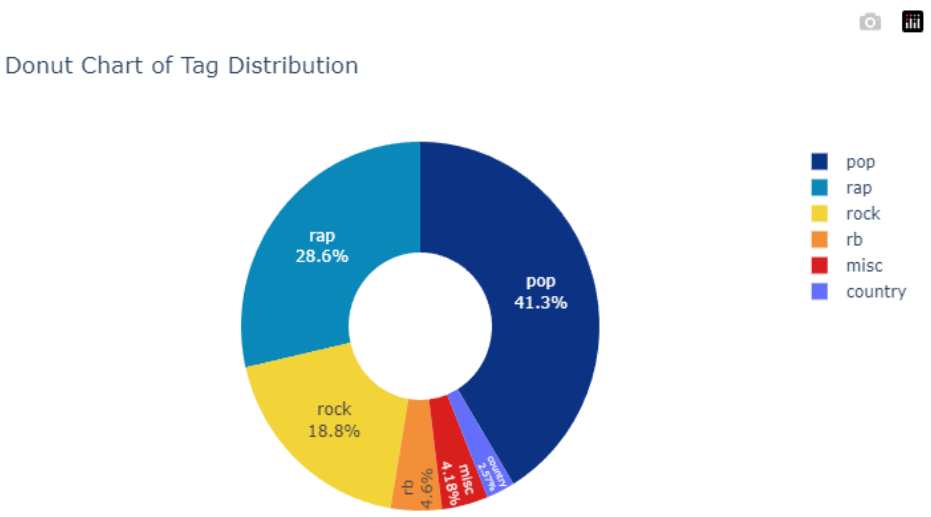


*Figure 1 - Distribution of tag variable*

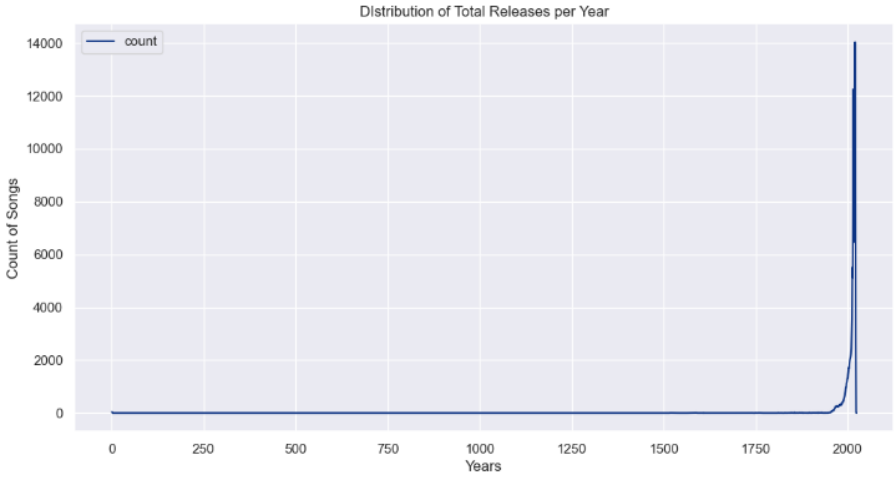


*Figure 2 - Distribution of Releases Per Year*

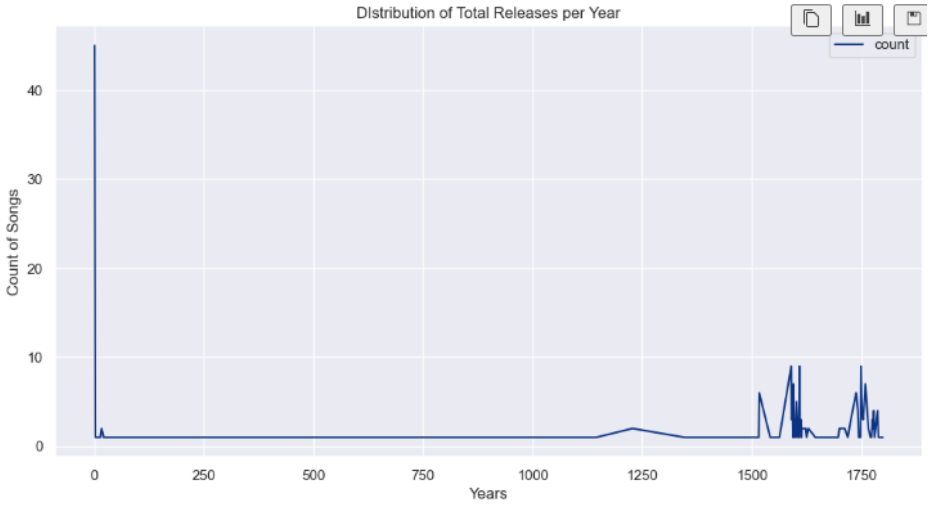


*Figure 3 - Distribution of releases in early years of humanity*

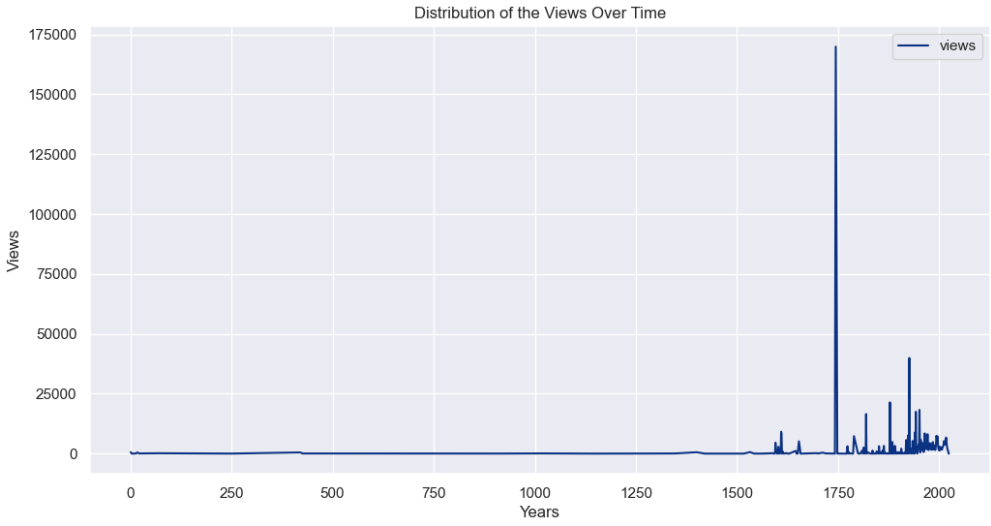


*Figure 4 - Distribution of Views Over Time*

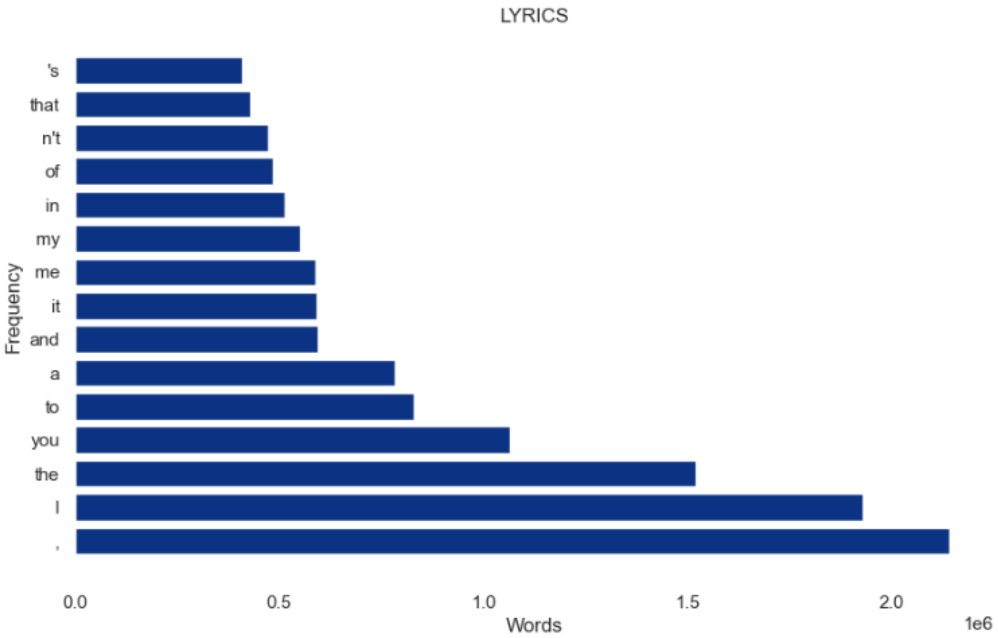


*Figure 5 - most frequent terms prior to pre-processing*

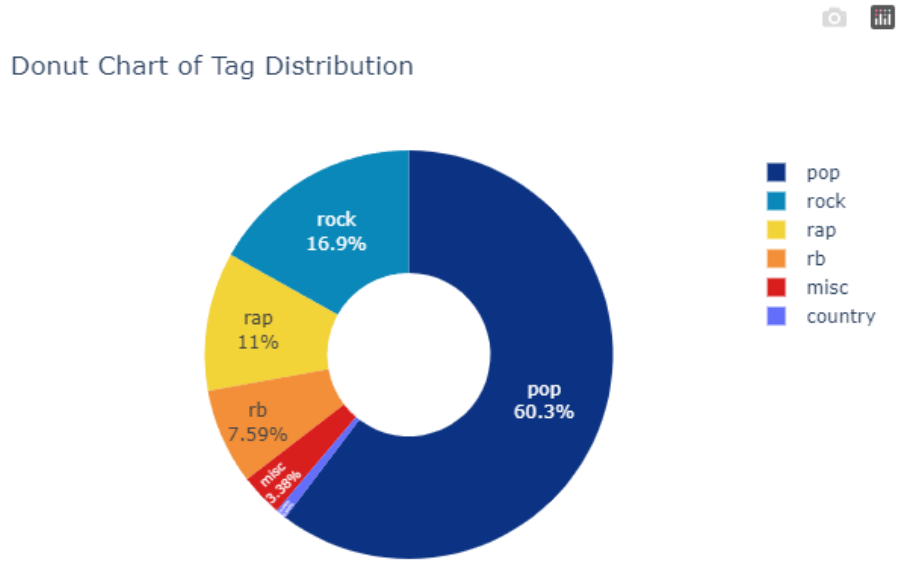


*Figure 6 - distribution of languages of non-english rows*

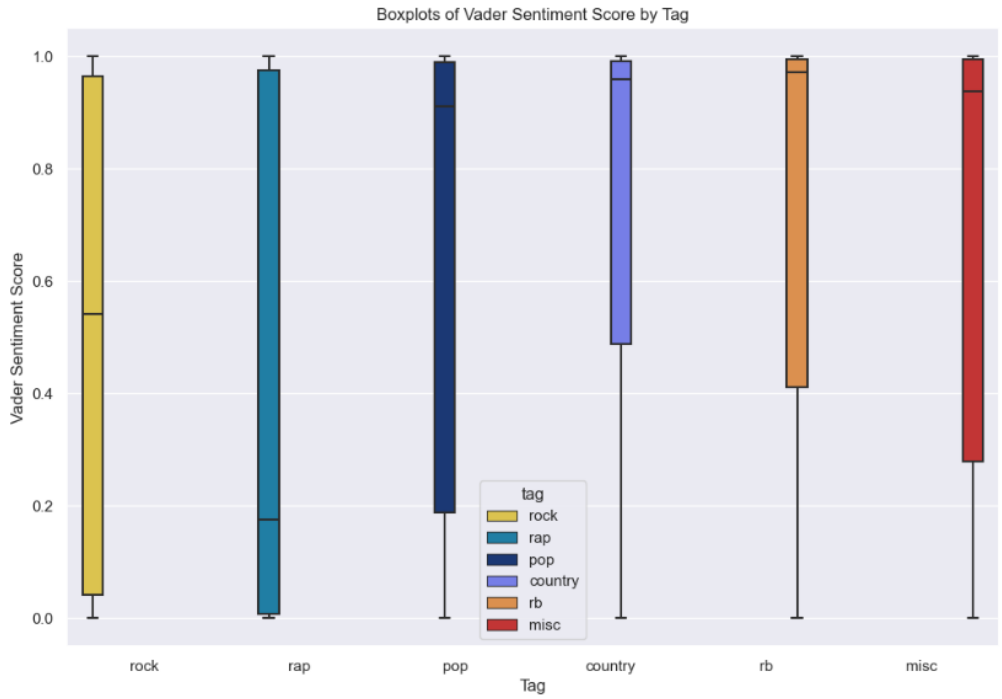


*Figure 7 – Boxplots of Vader Sentiment Score by Tag*

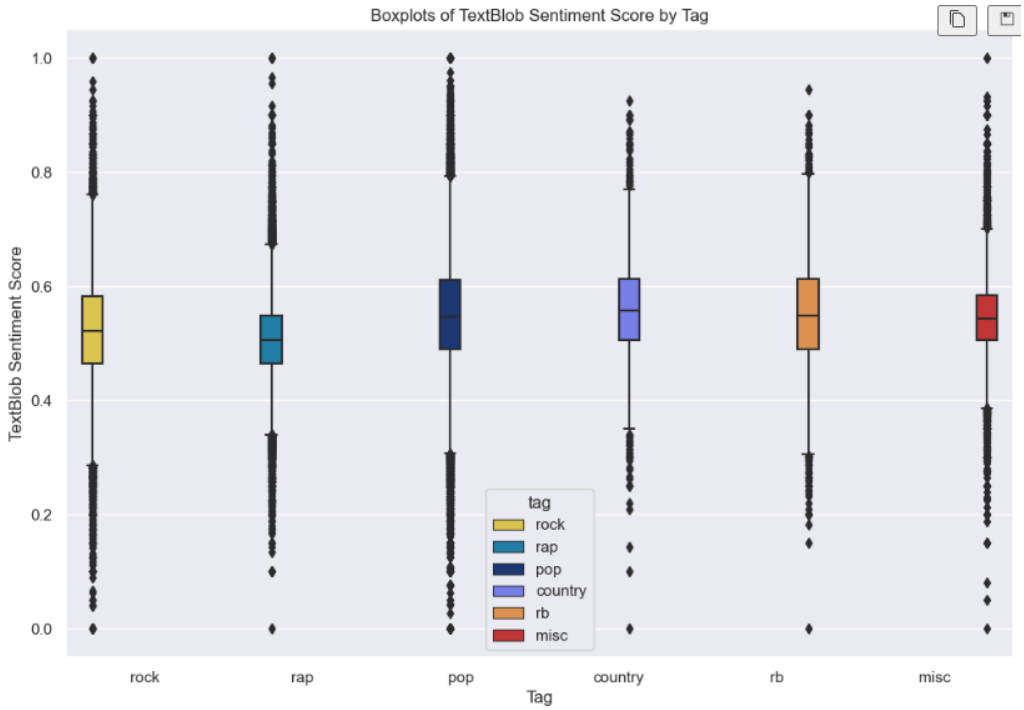


*Figure 8 - Boxplots of TextBlob Sentiment Score by Tag*

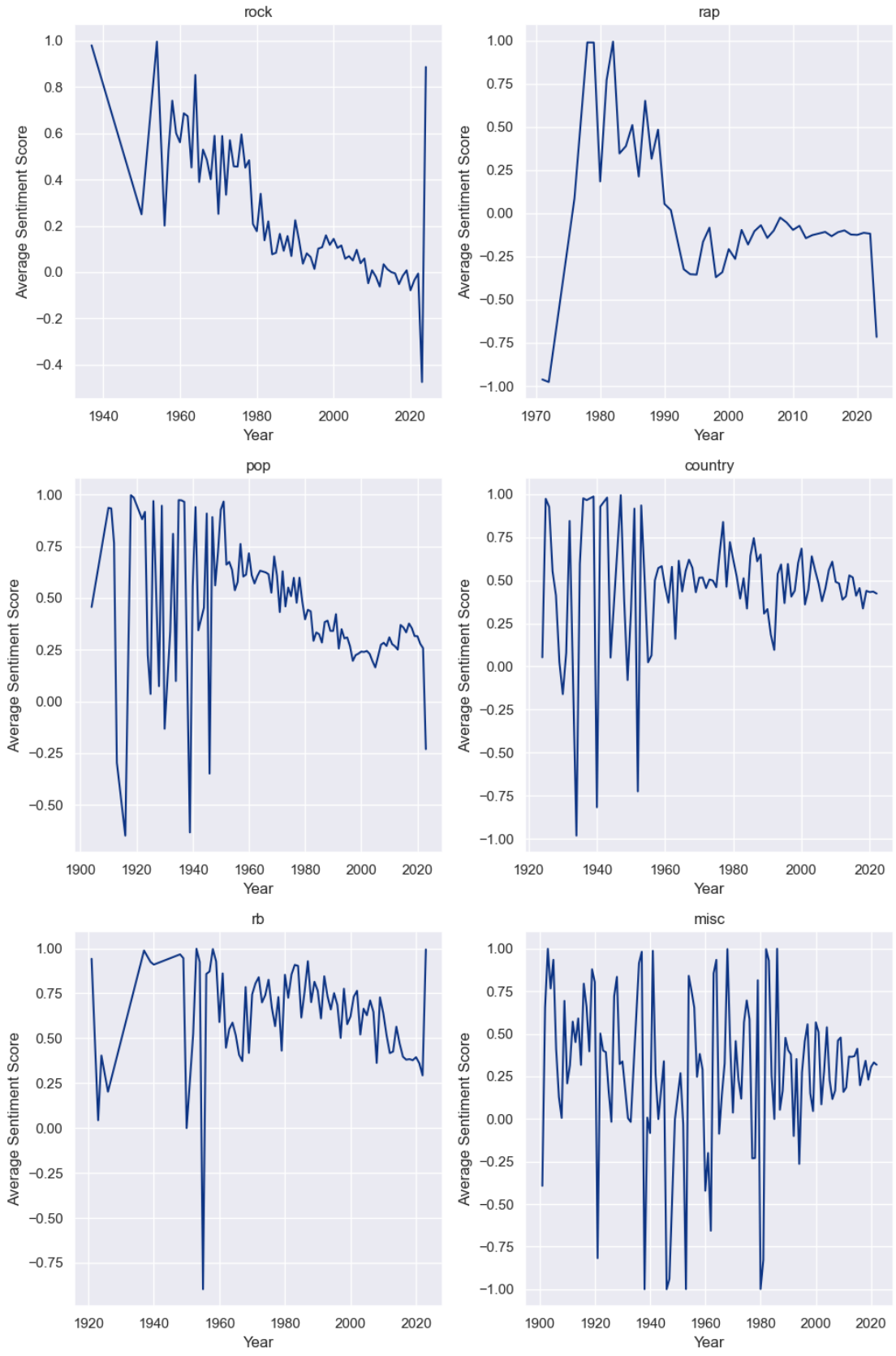


*Figure 9 - Average Sentiment Score Through the Years*

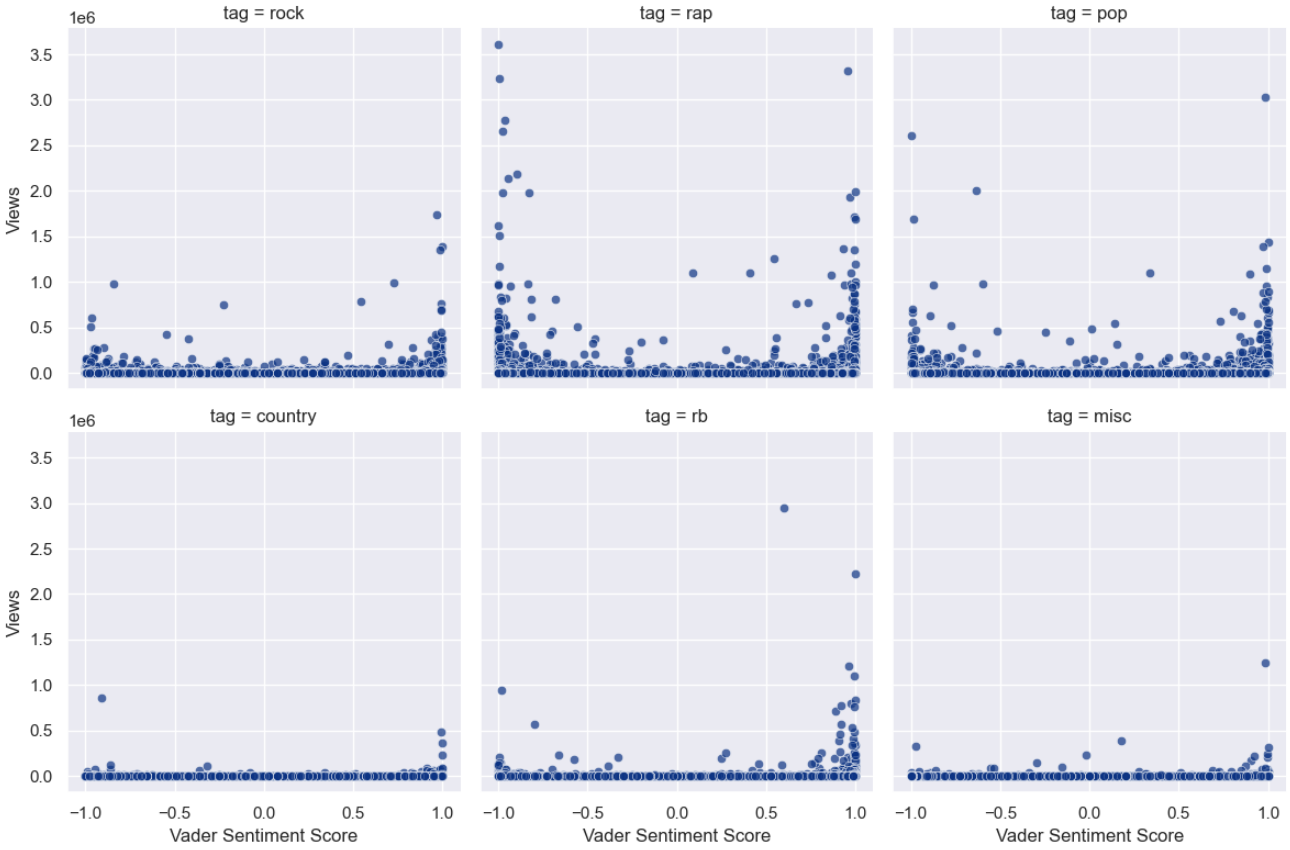


*Figure 11 - Vader Sentiment Score Variation with Views*

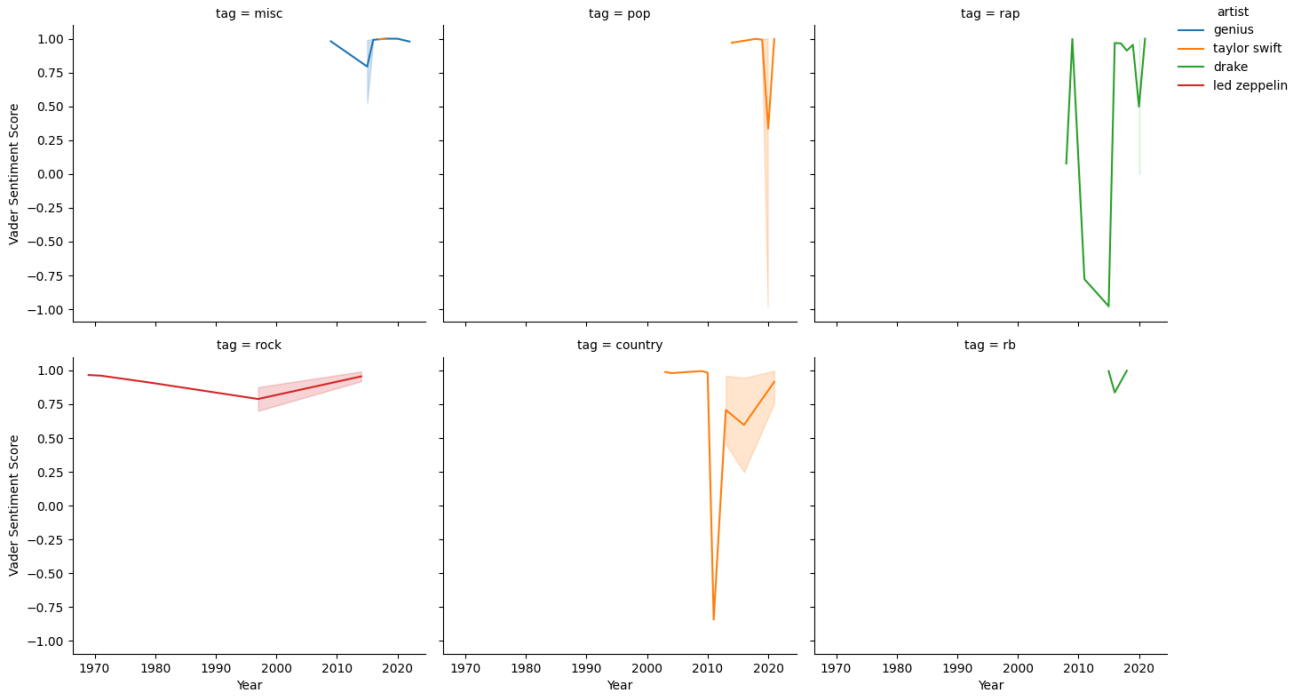


*Figure 10 - Variation of Vader score through years by artist*

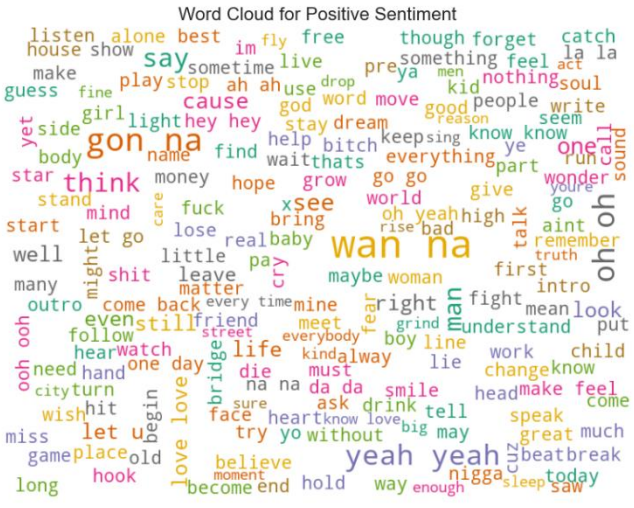


*Figure 13: WordCloud For Positive Vader Sentiment*


*Figure 12: WordCloud for Negative Vader Sentiment*


*Figure 14: WordClouds for emotions*

SelectKBest: feature selection filter method for selecting a k number of best features based on a scoring function. Evaluating the statistical significance (through an evaluation metric) of the relationship between each feature and the target. [13]

Support Vector Classifier (SVC): The fundamental goal of SVC is the maximization of the distance between two different classes. The first concept related with SVC is called the separating hyperplane and it consists of grouping the observations according to the values of the target in clusters. Then, according to how many dimensions there are, a point, line or plane is defined to separate those clusters. Briefly explained, the "maximum-margin hyperplane" suggests that the line (hyperplane) that separates both classes should have a margin that is merely the distance between the line and the nearest vector, which in turn becomes the "maximum-margin separating hyperplane". With this concept, SVM maximizes its ability to predict new and unseen observations. Another important concept in this algorithm is the soft margin, since not all data can be divided clearly with a straight line, the is a need for some "misclassifications". The soft margin allows *k* observations to be grouped on the wrong side of the hyperplane (considered outliers). Finally, the last concept to have in mind is the kernel function, which allows the algorithm to work on a low-dimensional set as it would on a set with higher dimension. However, it is important to have in mind that too many dimensions will lead to overfitting. [13]

Furthermore, the Linear SVC model was also used in the development of this project, the main difference between the two lies in loss functions used by default by each model.

XGB Classifier: This model builds sets of decision trees (weak learners), where each one of them corrects the mistakes of its predecessor, to produce a strong predictive model[15]. Its implementation is used for both classification and regression tasks.

Decision Trees: This model predicts the value of a target variable through the learning of simple rules gathered from the features of the dataset. It initiates with a base node, branches off based on feature values, and ends in leaf nodes - predictions.

Random Forest: This is an ensemble learning model that works through building several decision trees at training time and outputting the class that is the mode of the classes for classification, or mean prediction for regression[16].

Flair Sentiment Analysis: Flair is a NLP library developed by Zalando Research in Berlin, Germany that uses distilBERT embbedings and was trained on movies and product reviews from Amazon review corpus and that has shown an accuracy of 96.83, according to its authors, in the version used in this report. This version is a little less accurate than the transformer-based one, but much faster since it uses an RNN-based approach to perform the word embeddings. Essentially it takes the text, runs it into a bidirectional LSTM network and the final hidden layers are combined to create a representation of the sentence, vectoring it according to context. This vector is then introduced into a neural network with softmax activation in the last layer that predicts the probability distribution of the sentiment scores (positive or negative) and the highest probability corresponds to the chosen label. [8]