Final Project Report: Mood Classification in Song Lyrics

1. Introduction

This project aims to analyse song lyrics using natural language processing techniques to predict the mood (sentiment) of songs based on their lyrics. By leveraging machine learning models and data, this project explores the intersection of music and data science.

2. Problem Statement and Business Case

The objective is to build a model that can predict the mood (happy, angry, sad, relaxed) of a song based on its lyrics. This capability has practical applications in music recommendation systems, sentiment analysis in media content, and understanding emotional trends in music over time.

3. Data Collection

Data Sources:

Two distinct datasets were considered for training the models. The first dataset was downloaded from MoodyLyrics4Q, a dataset containing 2000 songs categorised into four moods: Happy, Angry, Sad, Relaxed. (more details on how the dataset was created can be found here: *Çano, Erion; Morisio, Maurizio. Music Mood Dataset Creation Based on Last.fm Tags. In: Fourth International Conference on Artificial Intelligence and Applications, AIAP 2017, Vienna Austria, 27-28 May 2017, pp. 15-26, DOI:10.5121/csit.2017.70603*).

The second dataset follows the same structure as the previous, Dataset-AllMusic-771Lyrics, containing 771 songs categorised for the same moods.

The moods in both datasets were a result of previous studies and ML models applied to determine the sentiment of specific songs, taking into consideration Russell's model of emotions.

Both models contain an index, song title, artist and mood. The lyrics needed for this current project were available in Musixmatch API.

Data Cleaning and Integration:

Even though the datasets had the same information, the column names were different, and the mood/classification of emotions had different values, mapped as follows:

- Q1 - Happy
- Q2 - Angry
- Q3 - Sad
- Q4 - Relaxed

For all datasets, the columns were updated to have the same denomination and the emotions were updated to have the same values, based on the mapping available with the datasets. Also for data cleaning, all the double spaces and "_" were replaced with a single space.

The sets were concatenated and the data cleaning was done for the new dataset created. All the duplicated rows and rows with NaN values were removed. The mood column, containing categorical data, was converted to numerical (using *LabelEnconder*), encoding needed for machine learning.

The lyrics were obtained from Musixmatch via api connection (using the artist and song title as input to search for lyrics). As not all of the songs had lyrics available, the rows with NaN values for lyrics were also removed.

The sum of both original datasets contained 2771 songs. After all the data cleaning, 9% of the songs were removed and the final dataset had 2522 songs with lyrics.

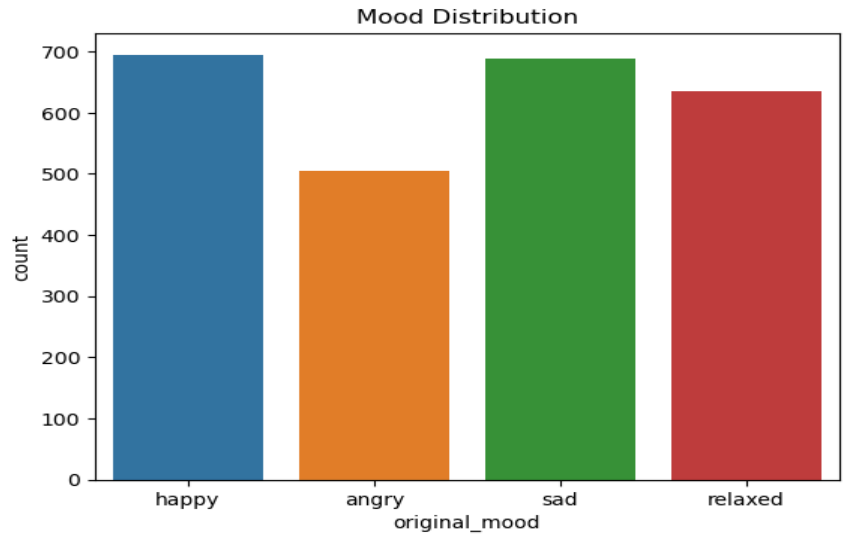
4. Exploratory Data Analysis (EDA)

After merging both datasets and having the lyrics, it was conducted an exploratory data analysis. The data has only 5 variables: *song title*, *artist* and *lyrics*, which are unique, *mood_original*, which is the original categorical variable containing the mood of the song, and *mood*, which is the encoding of mood_original and is the only integer variable in the dataset

At this stage, the preprocessing of text was not done, so, for conclusive results, the columns that still may have Nan values for lyrics were removed and the text was cleaned to remove a disclaimer that is at the end of the text to not impact on the metrics.Also, new columns were added with a word count and the average word length for each lyric.

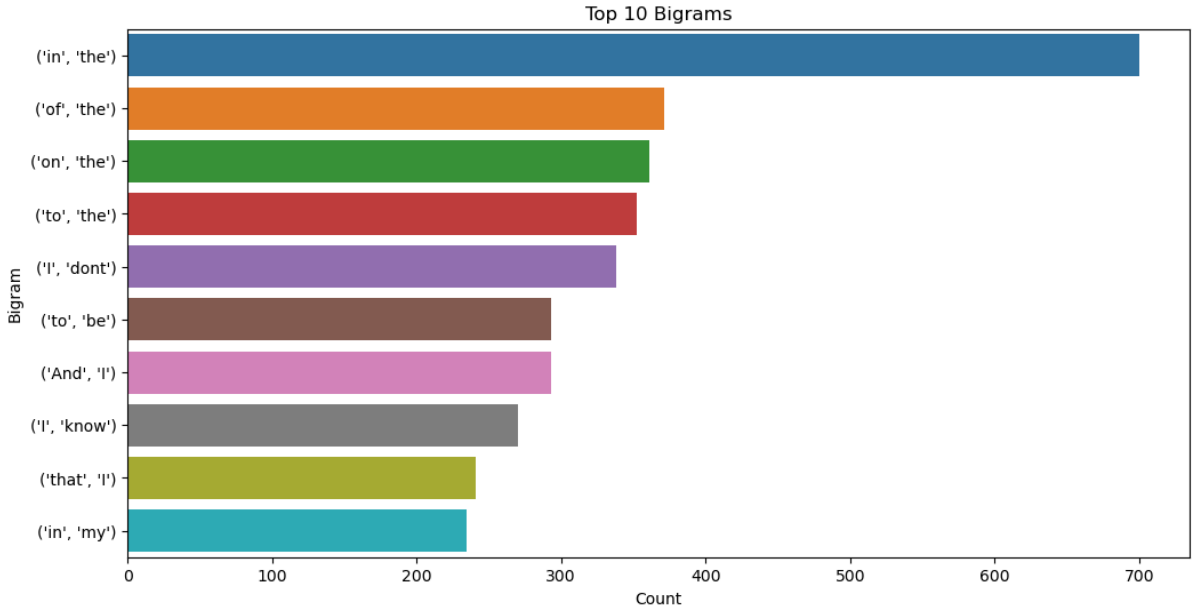
The statistical summary for this dataset has data for the only numerical value, but as this was encoded from a categorical one, the results are not very meaningful.

The moods distribution is balanced for the dataset, showing that our sample is good for training purposes. The mood with less songs is *angry*, but it still has enough data for the purpose.

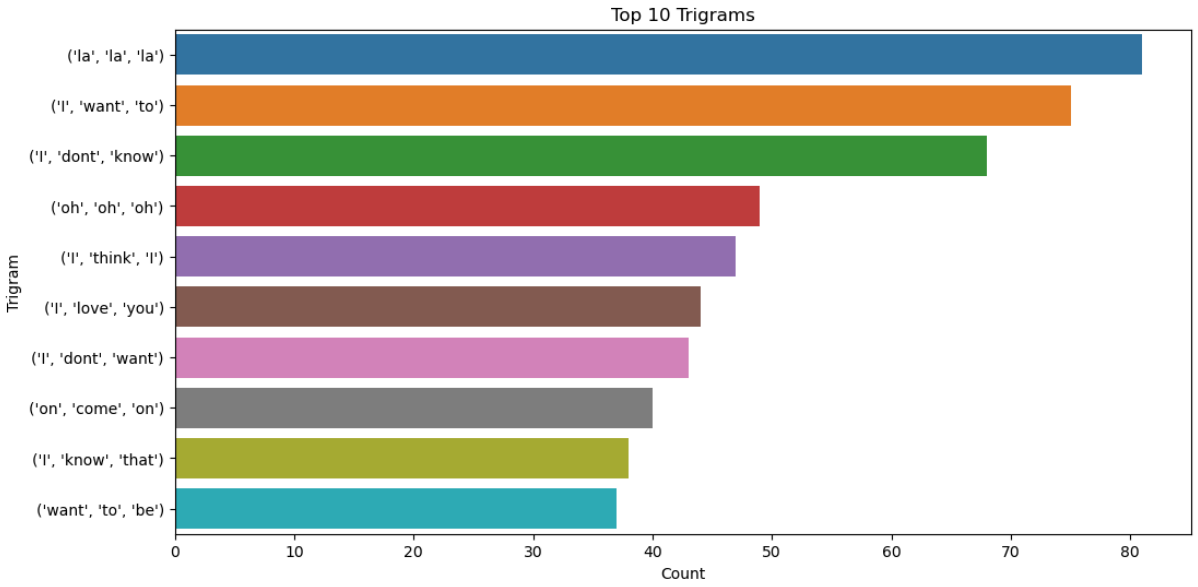


The n-grams were analysed as they may be helpful to gain insights into the most common topics and themes. In the context of NLP (Natural Language Processing), N-grams are typically contiguous sentences of words or characters. For this analysis, the N-grams considered were the bigrams and trigrams.

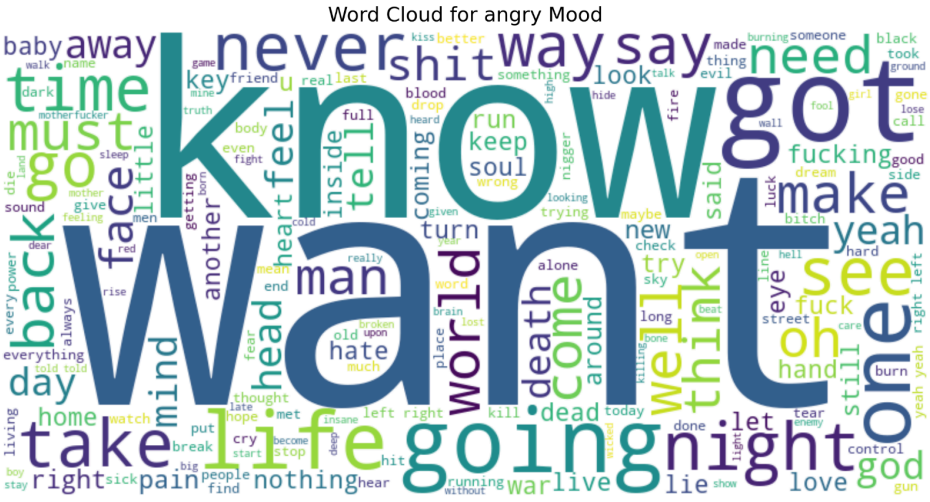
The top 10 bigrams show sequences as “in the”, “of the”, which are stopwords and will be removed in the text pre-processing as they are not meaningful to determine the mood.



The top 10 trigrams show sequences as ”la la la”, “oh oh oh”, repetitions of the same word that don’t have meaning, or sequences as “I want to” and “I don’t know”

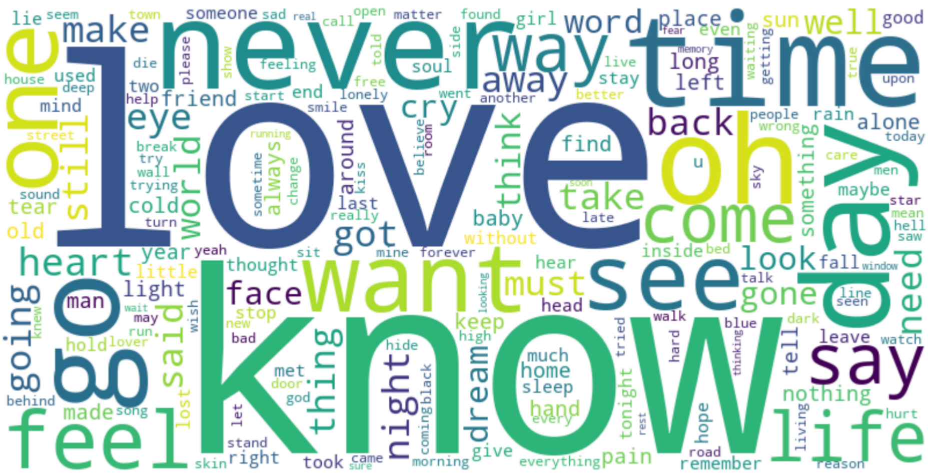


Lastly, were calculated the bags of words for each mood, to visually quantify the frequency of words in the lyrics for each mood. The word *love* is one of the most common for three of the moods, followed by word *know*.



Word Cloud for happy Mood

Word Cloud for sad Mood



Word Cloud for relaxed Mood

5. Text Preprocessing and Embedding

Text Preprocessing:

Some NLP techniques for data cleaning needed to be applied to the extracted lyrics. All the lyrics ended with the disclaimer “******* This Lyrics is NOT for Commercial us ******* (numerical code)”, which was removed. Also, a validation was done to check if the lyrics are in english and remove if they don’t (by calculating the ratio of words that match with the NLTK data for words in english). The lyrics were reviewed to handle contractions and expand the text (words like “don’t, can’t, etc” can be missed by the lemmatizer) and using *TextBlob* to correct the spelling mistakes that sometimes are found in lyrics. Finally, lyrics text was also preprocessed to remove numbers, punctuation, line breaks, lemmatize, remove stopwords and tokenize the text.

Embedding with OpenAI API:

Embeddings are vectors created by machine learning models for the purpose of capturing meaningful data about each object. These are important because they make it possible for computers to understand the relationships between words and other objects.

For this, the lyrics were splitted into smaller chunks (e.g., sentences or verses), as the embedding models usually have a maximum sequence length they can handle.

OpenAI's API was used to get embeddings for each chunk. The model used on OpenAI was the *text-embedding-ada-002* model, which has a maximum input length of 8191 tokens, another reason why it was important to split the lyrics into smaller chunks.

As the embeddings returned by the OpenAI API are high-dimensional (1536-D for *text-embedding-ada-002*), it was decided to reduce their dimensionality before using them in a machine learning model using PCA technique for dimensionality reduction.

This code includes a new function split_lyrics that splits the lyrics into smaller chunks if they are longer than the maximum input length of the embedding model. It also includes a step that calculates the mean of the embeddings for each chunk to get a single embedding for each lyric.

After generating the embeddings, the code uses PCA (Principal Component Analysis) to reduce their dimensionality. Reducing dimensionality using PCA is a common practice in data processing and analysis, especially when dealing with high-dimensional data like text embeddings has it helps in filtering out the noise and redundancy in the data by focusing on the most significant components and helps reducing the number of dimensions, improving the performance of the models and reducing the risk of overfitting.

6. Machine Learning Model Selection and Evaluation

Model Selection:

After some research for the best models for sentiment analysis, it was decided to consider the models listed below. These models were selected for their distinct strengths, collectively providing a comprehensive approach to sentiment analysis in song lyrics by combining traditional machine learning, advanced contextual understanding, and generative text capabilities.

Support Vector Machine (SVM)

This model was chosen as a baseline model because it is a robust and well-established machine learning algorithm that serves as an excellent starting point for classification tasks. It is relatively simple to implement and computationally efficient, making it suitable for initial performance benchmarks. Additionally, SVMs are effective in handling high-dimensional spaces like text data, especially with proper feature extraction techniques.

BERT (Bidirectional Encoder Representations from Transformers)

This model was selected for its ability to understand the context of a word in a sentence by considering the bidirectional context, which is crucial for accurately capturing the nuances in song lyrics. Leveraging a pre-trained BERT model provides a strong foundation due to its extensive training on a large corpus of text, leading to improved performance with less data. BERT has been proven to achieve state-of-the-art results in various NLP tasks, making it a reliable choice for sentiment analysis.

GPT-2 (Generative Pre-trained Transformer 2)

This model was included for its generative capabilities, which allow it to understand and predict sequences of text, particularly useful in generating and understanding lyrical content. Similar to BERT, GPT-2 is built on the transformer architecture but focuses on generating contextually relevant text, enhancing its ability to capture sentiment. GPT-2 can be fine-tuned for specific tasks such as sentiment analysis while leveraging its ability to generate human-like text, offering unique insights and applications.

Model Building and Evaluation:

The steps for building the models and evaluating them include loading and preparing the data, training the models with hyperparameter optimization, evaluating their performance, and saving the trained models and their evaluation reports.

Support Vector Machine (SVM)

Model Building

The SVM was chosen as a baseline model. It was implemented using SVC from scikit-learn. The training data features were the reduced embeddings from the lyrics, and the labels were the corresponding moods.

Hyperparameter Search

Hyperparameter tuning was performed using GridSearchCV with the following parameters:

- Kernel: ['linear', 'rbf']
- C: [0.1, 1, 10]

The best model was selected based on the highest accuracy obtained during the cross-validation.

Evaluation

The best SVM model was evaluated on the test set, and its performance was measured using accuracy and a classification report which includes precision, recall, and F1-score.

BERT

Model Building

BERT (Bidirectional Encoder Representations from Transformers) was used for its contextual understanding of text. The BertForSequenceClassification model from the transformers library was employed. The lyrics were tokenized and encoded using BertTokenizer.

Hyperparameter Search

Hyperparameter tuning was conducted using Optuna. The following hyperparameters were optimised:

- Number of training epochs
- Batch sizes for training and evaluation
- Warmup steps
- Weight decay
- Learning rate

Optuna's pruning callback was used for early stopping based on evaluation loss.

Evaluation

The best hyperparameters were used to train the final BERT model. The trained model was evaluated on the test set, and its performance was recorded in terms of accuracy and a classification report.

GPT-2

Model Building

GPT-2 (Generative Pre-trained Transformer 2) was utilised for its generative capabilities. The *GPT2Model* was paired with a custom classifier head to predict the sentiment. Lyrics were tokenized and encoded using *GPT2Tokenizer*.

Hyperparameter Search

Hyperparameter tuning for GPT-2 is more complex since it doesn't have a direct method for hyperparameter search like BERT's Trainer. It was manually implemented with a simple grid search for key hyperparameters such as learning rate and batch size. Hyperparameter optimization was done using Optuna, tuning the following:

- Learning rate
- Batch size
- Number of epochs
- Maximum sequence length

The training involved iterating through the training data with the optimised parameters and adjusting the model weights using the AdamW optimizer.

Evaluation

The final GPT-2 model was evaluated on the test set. The evaluation metrics included accuracy and a detailed classification report.

Each model was saved along with its evaluation report, ensuring the results could be referenced and the models could be reused or further refined.

7. Results and Discussion

Below is a detailed comparison of performance of the three different models for sentiment analysis on song lyrics: Support Vector Machine (SVM), BERT, and GPT-2, evaluation metrics, and a discussion on their strengths and limitations.

Evaluation Metrics

The models were evaluated based on the following metrics:

- Accuracy: The ratio of correctly predicted instances to the total instances.
- Precision: The ratio of true positive predictions to the total predicted positives.
- Recall: The ratio of true positive predictions to the total actual positives.
- F1-Score: The harmonic mean of precision and recall.

Performance Results

SVM Performance

- Accuracy: 0.593

- Classification Report:

SVM Classification Report:

	precision	recall	f1-score	support
0	0.65	0.60	0.62	98
1	0.63	0.69	0.66	131
2	0.60	0.50	0.54	121
3	0.51	0.58	0.55	127
accuracy			0.59	477
macro avg	0.60	0.59	0.59	477
weighted avg	0.60	0.59	0.59	477

BERT Performance

- Accuracy: 0.810
- Classification Report:

BERT Accuracy: 0.8095637583892618

BERT Classification Report:

	precision	recall	f1-score	support
0	0.92	0.84	0.88	491
1	0.86	0.84	0.85	655
2	0.77	0.70	0.74	589
3	0.73	0.85	0.79	649
accuracy			0.81	2384
macro avg	0.82	0.81	0.81	2384
weighted avg	0.81	0.81	0.81	2384

GPT-2 Performance

- Accuracy: 0.966
- Classification Report:

GPT Classification Report:

	precision	recall	f1-score	support
0	0.99	0.98	0.99	491
1	0.98	0.99	0.98	655
2	0.92	0.97	0.94	589
3	0.98	0.93	0.96	649
accuracy			0.97	2384
macro avg	0.97	0.97	0.97	2384
weighted avg	0.97	0.97	0.97	2384

Comparison and Analysis

SVM

Strengths: Simple and quick to train, provides a good baseline for comparison.

Limitations: Lower accuracy and F1-scores compared to more sophisticated models. Struggles with capturing complex patterns and context in lyrics.

BERT

Strengths: Strong contextual understanding, significantly higher accuracy and F1-scores than SVM. Good balance across all metrics.

Limitations: Requires more computational resources and time to fine-tune. Performance may vary depending on hyperparameter selection and dataset specifics.

GPT-2

Strengths: Highest accuracy and F1-scores among the three models. Excellent generative capabilities that allow for nuanced mood prediction.

Limitations: Highly resource-intensive to train and fine-tune. Complexity in setting up and optimising the model.

Best Model Identification

GPT-2 is identified as the best model based on its performance metrics:

- Accuracy: 0.966, which is the highest among the three models.
- Precision, Recall, F1-Score: Consistently high across all classes, indicating strong performance in both identifying and classifying different moods accurately.

GPT-2's ability to capture complex patterns in text and its generative capabilities provide a distinct advantage in mood prediction from song lyrics. Despite the higher computational cost, its superior performance justifies its selection as the best model for this task.

BERT also performs well, balancing contextual understanding with reasonable computational demands.

SVM, while less effective, provides a useful baseline and highlights the advancements brought by more modern NLP models.

8. Deployment and Practical Application

Using Flask, it was developed an app that takes song title and artist, fetches lyrics from Musixmatch, predicts mood using the best-performing model, and suggests similar songs. The app can receive three types of inputs from the user: the song/title of the artist, a piece of text from the lyrics, or the mood. When looking for the lyrics, the app will look for the string input on Musixmatch API and retrieve the first result, and will get the corresponding song to predict the mood and return five suggestions of songs to play. When searching for the mood,

the app will not apply the trained model and will just return five songs from the dataset with that mood.

Potential applications include music streaming services for personalised recommendations based on mood.

9. Conclusion

This project successfully demonstrated the application of natural language processing (NLP) techniques to predict the mood of songs based on their lyrics. By leveraging advanced machine learning models such as Support Vector Machines (SVM), BERT, and GPT-2, the project showcased the ability to accurately classify songs into one of four mood categories: happy, angry, sad, and relaxed. The process involved extensive data collection, preprocessing, and exploratory data analysis (EDA) to ensure a robust and clean dataset. Text preprocessing and embedding techniques, including the use of OpenAI's API, enabled effective feature extraction and dimensionality reduction.

The evaluation of different models revealed that while SVM provided a solid baseline, more sophisticated models like BERT and GPT-2 significantly outperformed it. Specifically, GPT-2 emerged as the best model, achieving an impressive accuracy of 0.966, demonstrating its superior capability in capturing complex patterns in song lyrics. The deployment of the model through a Flask application further illustrated the practical applications of this work, providing a user-friendly interface for mood-based song recommendations.

10. Future Work

Model Refinement and Hyperparameter Tuning:

- Continuously refine and tune the hyperparameters of the models to further improve accuracy and performance.
- Explore other advanced NLP models and architectures that may offer better performance or efficiency.

Expand Dataset:

- Incorporate additional datasets from various genres, languages, and time periods to enhance the model’s generalizability and robustness.
- Continuously update the dataset to include new songs and emerging trends in music.

Enhance Text Preprocessing:

- Improve the text preprocessing pipeline by incorporating more sophisticated techniques for handling misspellings, slang, and colloquialisms commonly found in song lyrics.

- Investigate the impact of different text normalisation methods on model performance.

Feature Engineering:

- Explore additional features such as acoustic properties, metadata, and user interactions to complement the lyrical analysis.
- Incorporate sentiment scores and emotional tone analysis to provide a richer feature set for the models.

Model Interpretability:

- Develop methods to interpret and visualise the model's predictions, providing insights into which parts of the lyrics influence mood classification the most.
- Implement explainable AI techniques to make the model's decisions more transparent and understandable to users.

Real-Time Prediction and Scalability:

- Optimise the deployed Flask application for real-time prediction capabilities, ensuring it can handle high traffic and large volumes of requests.
- Investigate the use of cloud-based solutions for scalability and enhanced performance.

User Personalization:

- Enhance the recommendation system by incorporating user preferences and listening history to provide more personalised suggestions.
- Implement collaborative filtering and content-based filtering techniques to improve the accuracy and relevance of recommendations.

Cross-Language and Cross-Cultural Analysis:

- Extend the model to support multilingual lyrics, considering the nuances and cultural differences in music sentiment across different languages and regions.
- Conduct cross-cultural studies to understand how mood in music varies globally and incorporate these insights into the model.

By addressing these areas, the project can further advance the state-of-the-art in mood classification in song lyrics, providing valuable tools for music recommendation systems, sentiment analysis, and understanding emotional trends in music. The continuous evolution and refinement of this work have the potential to significantly impact the music industry, offering deeper insights and enhanced user experiences.

11. Appendix

Link for GitHub repository:
https://github.com/sofiaggoncalves/Music-recommendation-project
Link for tableau dashboard:
https://public.tableau.com/views/IH-final-project/Dashboard1?:language=en-US&publish=yes&:sid=&:display_count=n&:origin=viz_share_link