Song mood detection

Yi Chen (yc2455), Yaxuan Huang (yh798),
Guangwei Jiang (gj94), Anqi Dong (ad829), Zhilin Li (zl635)

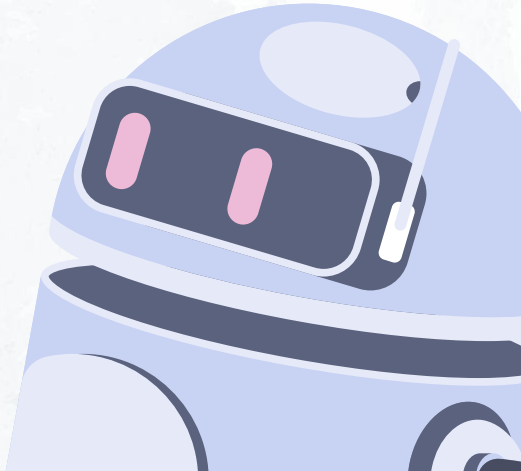


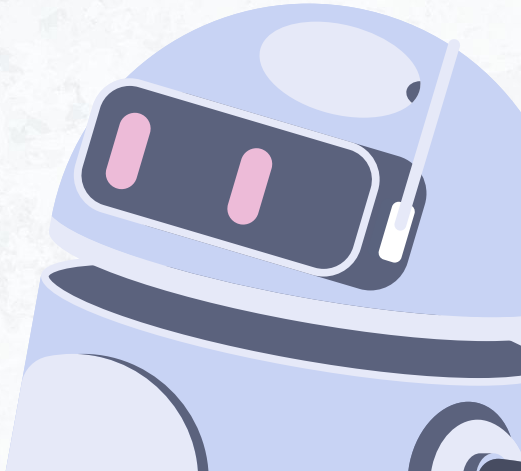
Table of contents

01 → Background

02 → Dataset

03 → Method

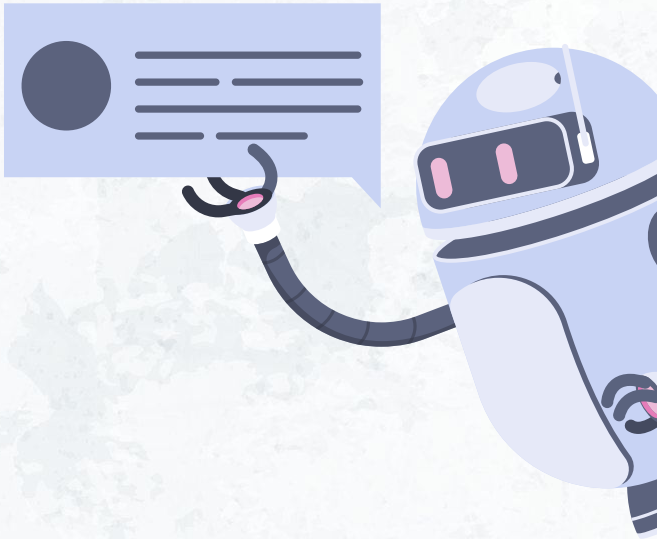
04 → Results & Discussion



Background

The project is to analyze song lyrics to determine the mood, aiming to develop methods for accurately attributing music emotions. The insights gained from the project can improve existing **recommendation systems**, enhance user experience, answer questions related to accurate mood classification, predict new song moods, and identify important features for mood classification.

Dataset-MoodyLyrics



(1) traits?

- Widely used
- Constructed based on MillionSongSubset
- 4 emotion categories: Happy, Angry, Sad, and Relaxed
- 2,595 songs

(2) challenge?

- copyright regulations, all song lyrics are removed. So we used the LyricsGenius, a Python library that allows developers to retrieve lyrics from the Genius.com database to collect the lyrics.

02 →

Methods

Data Cleaning

(1) Duplicate Data and Imbalance Issue

- Removed 86 duplicate entries and obtained a balanced dataset
- Discovered only 2405 unique lyrics out of 3085, because same song showed up in different playlist with same mood

(2) Lyrics Cleaning

- Removed all null values
- Initially removed punctuations, numbers, newline characters etc, but model was not performing satisfactorily
- We found a general pattern in lyrics structure which varied across songs
- So we removed all patterns through regex and further removed null values
- Encoded Count_vector and TF_IDF with cleaned lyrics for classification

Feature Extraction

We started by extracting features from our lyrics dataset. Steps taken: Bag of Words analysis, sentiment analysis, and word embedding.

(1) Word Embedding

We used GloVe to map each word to a high-dimensional vector and explore semantic relationships between words.

(2) Sentiment Analysis

We used textblob to extract sentiment polarity, lexical diversity, and average word length. This helped us better understand the emotional tone of our lyrics.

Models

(1) Supervised K-Means

- Computes centroids for each emotion class & assigning lyrics to the tags of the closest clusters.
- Split the dataset into a training set (90%) and a test set (10%)
- Evaluated the classification accuracy on that test set

(2) KNN（k-Nearest Neighbour）

- A generalization of the first model (K-Means)
- Evaluate our model for several different k values

  [1, 3, 5, 7, 9, 11, 13, 15, 17, 19, 21]

- Determine the best value of k from the results
- Compare the accuracy rate with and without preprocessing our lyrics

Models

(3) Random Forest

- An ensemble learning method that aggregates results from multiple decision trees, suitable for high-dimensional Data
- Used GridSearch for parameter tuning, gained the best parameter that max_depth=10, min_samples_leaf=2, min_samples_split=5, n_esitimators=2

(4) SVM (Support Vector Machine)

- Train a SVM using the algorithm available in sklearn
- First explore the model effect using sklearn default parameters
- Than, perform a Grid Search to properly tune our model
- Using a cross validation approach to evaluating accuracy.

Models

(5) Gradient Boost

- A machine learning technique for regression and classification problems
- Omit any grid search for parameters tuning
- Compare the results of SVM and Gradient Boost to determine which one to choose

(6) ANN (Artificial Neural Network)

- We try to train an artificial neural network model with two hidden layers.
- The output layer consists of 4 units, employing the 'softmax' activation function for multi-class classification.
- Compiled with the 'adam' optimizer, 'categorical_crossentropy' loss, and 'accuracy' metric

04 →

Results & Discussion

Result

Table 1. The comparison between models

method	accuracy	precision	recall	f1-score
KNN	0.76	0.77	0.76	0.75
Supervised K-Means	0.23	0.14	0.23	0.17
Gradient Boost	0.81	0.82	0.81	0.81
Random Forest	0.81	0.81	0.81	0.81
SVM	0.84	0.84	0.84	0.84
ANN	0.87	0.87	0.87	0.87

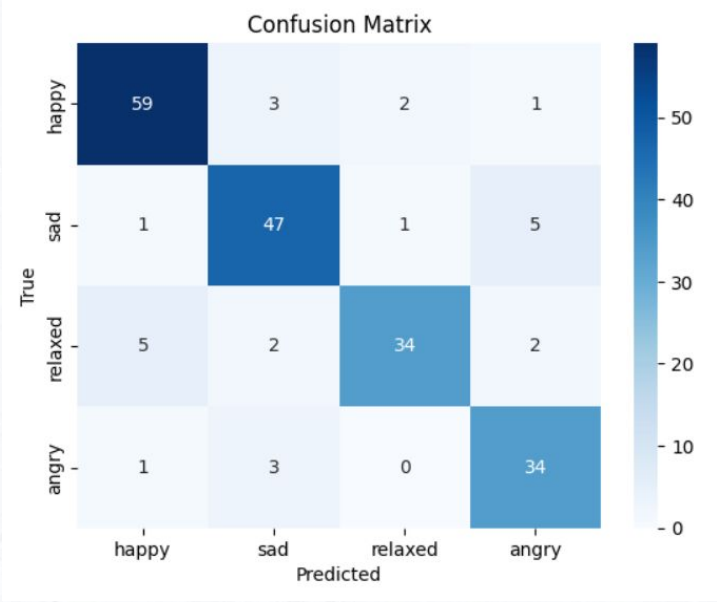


Fig 7. Confusion matrix for ANN model

Discussion

(-) Important of data clean ⟶

Through this project, we clearly realized that data cleaning has a huge impact on the efficiency and accuracy of training models. The data cleaning process proved to be crucial in removing noisy and irrelevant data points that could potentially lead to biased results in the sentiment detection model. It is through this meticulous approach to data preparation that we are able to fine-tune our algorithms and achieve higher levels of accuracy.

(-) Data and Model ⟶

Second, we found that the data needed to be consistent with the requirements of the model. During early stage of the task, the requirements for the dataset were not fully discussed, resulting in the dataset not being in the same format as required by the model, and further changes need be made to the dataset.

Future of Work

For future work, we hope to move away from relying on MoodyLyrics and instead rely on data obtained directly from Spotify. This will allow us to have access to more accurate and diverse information. By leveraging Spotify's extensive user base and algorithms, we can collect data on music preferences, trends and patterns. This new method of accessing data will give us a qualitative leap forward, of which we think the most important is the ability to fully align the data with real-world data.

Reference

[1]Korzeniowski, Filip, et al. "Mood classification using listening data." arXiv preprint arXiv:2010.11512 (2020).

[2]Chaturvedi, Vybhav, et al. "Music mood and human emotion recognition based on physiological signals: a systematic review." Multimedia Systems 28.1 (2022): 21-44.

[3]Akella, Revanth, and Teng-Sheng Moh. "Mood classification with lyrics and ConvNets." 2019 18th IEEE International Conference On Machine Learning And Applications (ICMLA). IEEE, 2019.

[4]Çano, Erion, and Maurizio Morisio. "Moodylyrics: A sentiment annotated lyrics dataset." Proceedings of the 2017 international conference on intelligent systems, metaheuristics & swarm intelligence. 2017.

Thanks!