CS361: Spotify Song Mood Analysis using Classification and Clustering

Achintya Gupta Ankita Kanoji Arnav Sharan Kannan Rustagi

Abstract

In this report, an innovative approach for modeling a track's mood based on audio components from the **Spotify Web application program interface (API)** available on the Spotify for Developers is used. We have mainly classified the song in one of four moods – **calm, happy, energetic, and sad**. We extracted features for each song from our playlist using the Spotify Web API and used some of the important features that helped in clustering songs according to different moods. For **EDA**, we have constructed Heatmaps correlating the features, Radar plots, and KDE plots for better visualization of our data which will indeed help in observing the data distribution in each class representing a specific mood. As the initial step, we have used **Gaussian Naive Bayes classifier** and **Random Forest** as our classification models. In Naive Bayes, we used **K-fold cross-validation** for evaluating the accuracy of our model.

1. Introduction

1.1. Motivation

"Can we teach a computer to learn how music will make people feel?" We currently use many different methods to classify the music that we listen to, such as genre, artist, etc. For us, however, a song usually induces a feeling in us that cannot be easily explained, but often we can associate certain moods with them. Identifying the mood of a song can improve the existing song recommendation algorithms, which only use the genre and similar artists.

1.2. Target Problem

We have a large collection of digital music, and while existing methods to categorize the songs are quite complex, taking into account various metrics, they often fall short in capturing the human emotions invoked by the music. Quantifying such data through statistical methods can prove to be quite a challenging task, and tackling this obstacle through Machine Learning methods is a worthwhile problem to solve.

1.3. Objective

The primary objective of this model is to accurately identify and categorize the mood associated with a user given song. Spotify is the largest music streaming platform in today's world. We focus on improving its capability of recommending a song based on the user's mood. If a user is listening to a song, our model will predict the mood associated with that song and spotify can recommend songs that come under the same mood category. This will highly improve user's engageability and will lead to increased satisfaction and loyalty. Only genre based and artist based suggestions may not resonate with the user's mood, thus integrating mood based suggestions will enhance precision and personalization in song recommendations.

2. Methods

2.1. Data Scraping

For data scraping, we have followed the given process-

- We created an account on the platform 'spotify for developers', followed by the creation of a dummy app, whose client credentials we will use to extract the required information.
- Further, we have used the spotipy library of python to help us extract the song features. This library ultimately helps us by making a call to the spotify web API which in turns returns the features of each song in the playlist given to it as input.
- One major challenge we encountered here is that there is a certain limit set at the number of API calls that will occur in a time frame of 30s by the dummy app we have created. Hence, we had to extract the information in batches of roughly this was a tedious process.
- Then, we ultimately combined the data into a csv file.

2.2. Data Collection

We had initially planned to make our own dataset using the spotipy python library which fetches the data from several playlists labelled as 'Calm', 'Happy', 'Sad' and 'Energetic. We were faced with the following issues-

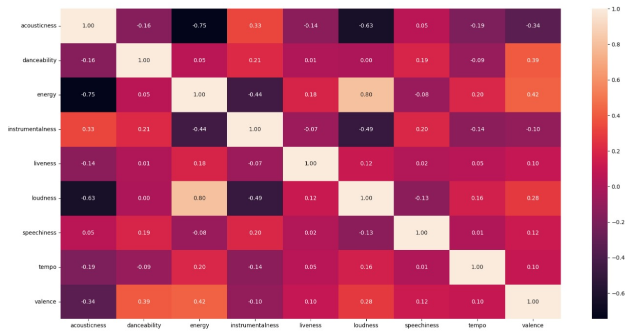
- The standard playlists made by Spotify which seemed to be accurate were fairly small (usually 50-100 songs) which did not provide for proper accurate training of the model.
- Large user made playlists had a lot of variance, where songs overlapped as well as could be incorrectly labelled based on the biased preferences of the user who created the playlist.
- We had to remove the duplicates, since our model does not handle multi-label classification. Furthermore, on plotting the data, we realised that the given playlists had roughly the same mean values for all the features. This might again be due to variety in Hindi and English songs and differences in how someone personally classifies different songs.
- This creates a need for manual labelling for data, which is not feasible for large datasets for the given timeline of the project.

Hence we went forward with a dataset found online.

2.3. Exploratory Data Analysis & Feature Selection

Firstly, we dropped those columns from the dataset which were not relevant for predicting the mood.

- We used a heatmap plot to be able to visualize the correlation between all the available features. We found that there is not much correlation in our feature set, hence we decided to use all of them as inputs to our model.



	acousticness	danceability	energy	instrumentalness	liveness	loudness	speechiness	tempo	valence
acousticness	1.00	-0.16	-0.75	0.33	-0.14	-0.63	0.05	-0.19	-0.34
danceability	-0.16	1.00	0.05	0.21	0.01	0.00	0.19	-0.09	0.39
energy	-0.75	0.05	1.00	-0.44	0.18	0.80	-0.08	0.20	0.42
instrumentalness	0.33	0.21	-0.44	1.00	-0.07	-0.49	0.20	-0.14	-0.10
liveness	-0.14	0.01	0.18	-0.07	1.00	0.12	0.02	0.05	0.10
loudness	-0.63	0.00	0.80	-0.49	0.12	1.00	-0.13	0.16	0.28
speechiness	0.05	0.19	-0.08	0.20	0.02	-0.13	1.00	0.01	0.12
tempo	-0.19	-0.09	0.20	-0.14	0.05	0.16	0.01	1.00	0.10
valence	-0.34	0.39	0.42	-0.10	0.10	0.28	0.12	0.10	1.00

Figure 1. Feature Correlation Heatmap

- Next, we made plots for how all the songs are distributed for each feature, i.e., the frequency of songs for different feature values.
- Building upon the previous step, we also made Kernel Density Plots(KDE plots), wherein we can visualise the data using a continuous probability curve, plotted separately for each feature. Then we also plotted class conditional distribution about the features for each class and every feature, wherein we could see how for most features it resembled the Gaussian distribution, this is why we decided to add Naive Bayes Classifier as a modelling method as well and see its performance.

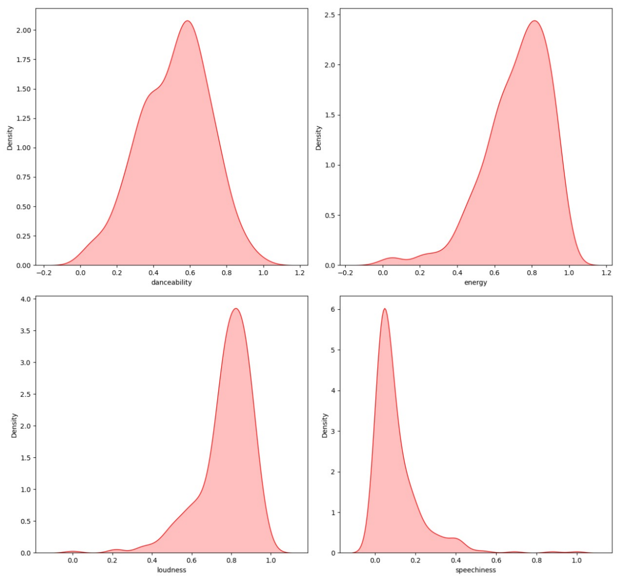


Figure 2. Class Conditional Density Plots (for Happy)

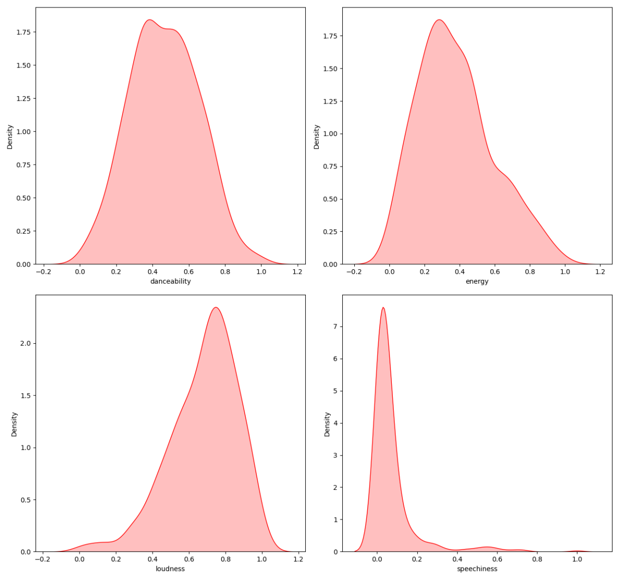


Figure 3. Class Conditional Density Plots (for Sad)

- For each feature, we have shown its average value for every class to see the differences and similarities between our defined classes in terms of the given features.

- We segregated the data based on classes. Then for each class, we first normalised each feature using min-max normalisation. Then, we took the mean and median values of all features and plotted a radar chart to visualize what average values all features take for each class and how they are different for all the classes. The radar charts based on mean and median were fairly the same for every class.

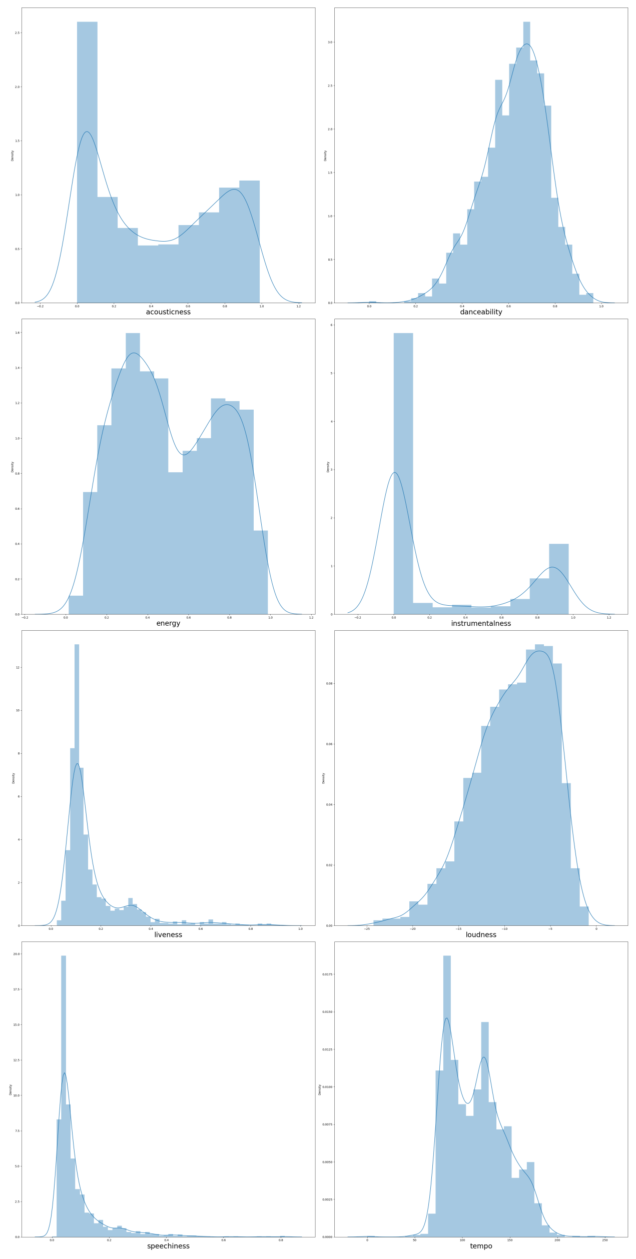


Figure 4. Frequency Distribution Graphs

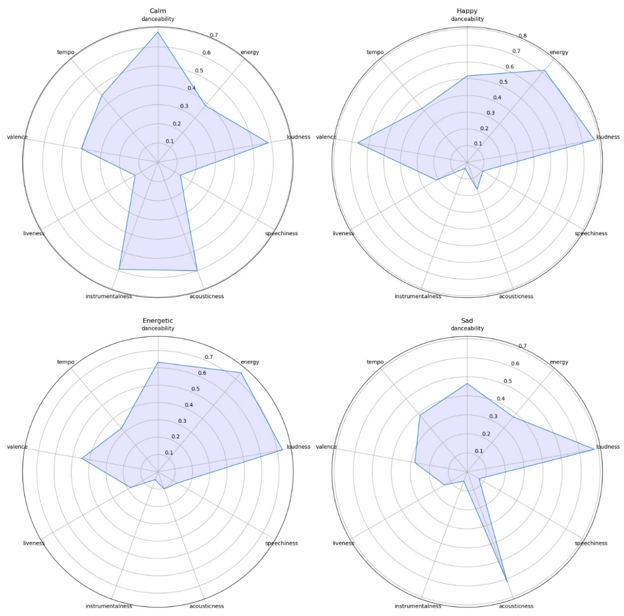


Figure 5. Radar Plots for each mood

2.4. Modelling Methods

There are 2 methods to go about the problem:-

2.4.1. Multi-class Classification:

One approach is to treat mood identification as a multi-class classification problem. We can use the mood categories as defined above, and train a classifier to predict the mood label of a given song.

We have implemented Random Forest algorithm for classification as it is robust to overfitting and noise which can be problematic while analysing an artistic field such as music. Moreover, it performs well on imbalanced data, for example, if we have fewer samples of a particular mood class. It can also model non-linear relations within features and mood labels.

To implement random forest, we have first implemented a Decision Tree class. DecisionTree constructs trees based on input data with stopping criteria like maximum depth and node purity. RandomForest builds an ensemble by training trees on random subsets of data (bootstrapping) and random subsets of features (feature bagging). During prediction, each tree independently predicts, and the final result is determined by majority voting. This implementation allows customization of parameters like tree count and depth.

We have implemented Gaussian Naive Bayes (GNB) for classification. This model is simple to implement, and it is highly scalable with the number of predictors and data

points, and can handle high-dimensional data well. Moreover, it gives the best results when our data points follow a Gaussian distribution, as seen in the KDE plots.
GNB Classifier assumes that predictors (features) are conditionally independent, and also that all features contribute equally to the outcome. It then computes the probability by multiplying the class-conditional probabilities for each feature with the prior probability for each class.
Further, we have used K-fold cross-validation to evaluate the accuracy of our model.

2.4.2. Clustering:

Another approach is to cluster songs based on similarities in their audio features. Instead of predefined mood categories, this method groups songs based on similar acoustic properties. By applying clustering algorithms, we can uncover patterns and similarities among songs that may indicate shared mood characteristics.

We can visualize these clusters and compare them with existing genre values present in the data to gain further insights into the relationship between musical characteristics, mood and genres of the songs.

We will use K-means for the clustering problem since it works efficiently for large dimensional feature sets such as the one for Spotify songs that we are using. It is also based on the concept of similarity and closeness between features, which can be expected based on Russell's Circumplex Model of Emotions in Figure(1).

3. Progress

We have completed Data Scraping and Collection. We have also completed Exploratory Data Analysis and Feature Extraction. Moreover, the two mentioned classification algorithms, Random Forest Classifier, and Gaussian Naive Bayes Classifier, were implemented and evaluated on two datasets. The first dataset was collected by scraping data from the Spotify playlists we had compiled from multiple resources online, while the second dataset was sourced from a repository on GitHub.

Performance Metrics:

- **Random Forest Classifier:**
  - Dataset 1 (Spotify playlists):
    - Custom Implementation: **42.254%**
    - sklearn Implementation: **38.623%**
  - Dataset 2 (GitHub dataset):
    - Custom Implementation: **71.477%**
    - sklearn Implementation: **72.325%**
- **Gaussian Naive Bayes Classifier (10-Fold Cross Validation) :**
  - Dataset 1 (Spotify playlists):
    - Custom Implementation: **43.555%**
    - sklearn Implementation: **42.232%**
  - Dataset 2 (GitHub dataset):
    - Custom Implementation: **71.180%**
    - sklearn Implementation: **71.218%**

Notebook link for reference:

- Google Drive

4. Conclusion

In our Spotify Song Mood Analysis, we have gained hands-on experience in Machine Learning from the very first step of Data Collection to Exploratory Data Analysis, to implementing ML Models from scratch. Despite initial challenges in obtaining labeled data, we tried out our own dataset as well as a dataset we found online. Through visualizations like heatmaps and radar charts, we gained insights into the distribution of audio features and their relationship to mood,

We have currently experimented with two classification models, namely, Gaussian Naive Bayes and Random Forest. Gaussian Naive Bayes performed fairly well due to the Gaussian nature of the class conditional distribution of the different features in the data. Random Forest, implemented with Gini impurity-based Decision Trees, also performed similarly and fairly good as we have implemented bootstrapping and feature bagging which helps to reduce variance and model the data accurately.

References

Moschitto, M. *GitHub Dataset*, 2022.

Nuzzolo, M. *Music Mood Classification (Medium Article)*, 2015.

Patra, B. G., Bandyopadhyay, S., and Das, D. *Unsupervised Approach to Hindi Music Mood Classification*, 2013.

Solano, H. D. *Clustering My Spotify Songs and My Mood Prediction*, 2022.

Wallach, J., Corr, B., and Moschitto, M. *Mood Classification of Spotify Songs*, 2021.