Music Mood Classification

CS 229 Project Report

Jose Padial
Ashish Goel

Introduction

The aim of the project was to develop a music mood classifier. There are many categories of mood into which songs may be classified, e.g. happy, sad, angry, brooding, calm, uplifting, etc. People listen to different kinds of music depending on their mood. The development of a framework for estimation of musical mood, robust to the tremendous variability of musical content across genres, artists, world regions and time periods, is an interesting and challenging problem with wide applications in the music industry.

In order to keep the problem simple, we considered two song moods: *Happy and Sad*.

Database

As with any learning project, the size and quality of the data set is key to success. We initially underestimated the difficulty in acquiring a music database labeled by mood. Building the labeled Happy/Sad database proved to be a challenging journey for a number of reasons, not the least of which being the difficulty in making the subjective decision to label songs as strictly 'Happy' or 'Sad'.

We began by analyzing songs from our personal music collection and soon realized the need for a larger and more comprehensive database. After spending some time searching for a suitable database, we found the Million Song Database (MSD), a freely-available collection of audio features and metadata for a million contemporary popular music tracks. The MSD was compiled by labROSA at Columbia University with the help of analysis done using Echo Nest API (an open source platform for analysis of audio files). Each track data file contains a wealth of tempo, mode (minor/major), key and local harmony information. This is the information we planned to extract ourselves via time-domain and spectral methods, and were thus very excited to find it in this database.

The entire database of a million songs is 300GB in size. Downloading and unpacking the database alone took several days, and crawling through the database within the timeframe of this project turned out to be an infeasible task. Hence, we largely operated with a subset of the database containing 10,000 songs.

The most challenging task was generating accurate Happy/Sad labels for the songs contained in this database. Tags from the website last.fm were available for the songs contained in the MSD. Out of the 1 million MSD songs, nearly 12,000 had a 'Happy' tag, and over 10,000 a 'Sad' tag. However, upon inspection of these songs, we discovered that the majority of Happy/Sad tags were incorrect.

Ultimately we hand-labeled songs from the 10,000 subset to generate our training set. The final data set comprised 137 sad songs and 86 happy songs. The drop from 10,000 to 223 is a result of most songs being unfamiliar to us, and many of those we knew are not clearly 'Happy' or 'Sad'.

Hold-out cross validation was used for testing the performance of our learning algorithm. 70% of the final data set was used for training and 30% of it was used for testing purposes.

Feature Selection

The following were considered as candidate features for the classification process

- Tempo: the speed or pace of the piece, measured in beats-per-minute (BPM). This is a time domain feature which captures the rhythm of the song.
- Energy: obtained by integrating over the Power Spectral Density (PSD).
- Mode: indicates if a piece is played in major or minor key.
- Key: identifies which of the 12 keys the song has been played in (Fig. 1).
- Harmony: relative weighting between notes, characterized as chords or modes.

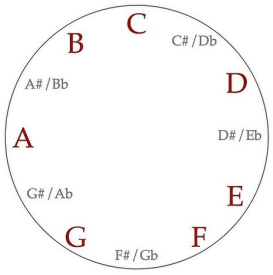

Figure 1: 12-note musical scale.

Harmony

While feature elements such as Tempo and Energy were easy to obtain and use, a lot of time and effort was spent on sensibly extracting the harmony information from the data. The MSD provided us with the PSD of 0.3 seconds long segments of the song arranged in 12 bins corresponding to the frequencies of the 12 different notes. Hence a song of duration 300 seconds was divided into 1000 segments, yielding a pitch matrix of size 12x1000 for each song.

This local harmony information could be processed and used in several ways. If we had a large enough training set (approximately 10 times the size of the feature vector), we could have simply passed the huge 1000x12 matrix into the classifier. However, since the data set was limited, we had to intelligently capture the harmony information in a small-sized feature vector. The need for doing this will be more evident from the learning curve analysis (Fig. 4) which shows that we were suffering from the problem of high variance. The motivation for the approach we adopted came from the concept of modern musical modes as shown below in Fig. 2.

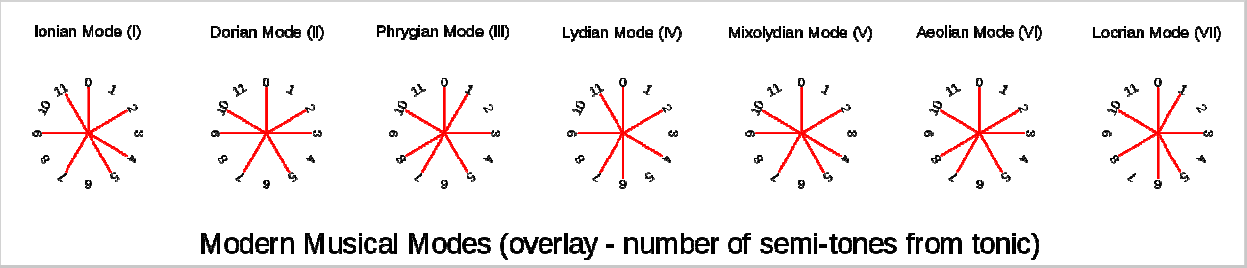

Figure 2: Musical modes, each corresponding to a 7-note subset of the total 12 musical notes.

We hypothesized that extracting the above modes from the harmony information would contribute to the mood detection significantly. Several attempts were made to associate the song with one of the 7 musical modes. We switched to the time domain and tried working over segments of different lengths but couldn't succeed in assigning a mode to a majority of the songs in our database. Eventually, we picked the 7 most important notes for each of the 0.3 seconds long segments, averaged over the entire song and subtracted the key from each of the notes to obtain a 7-dimensional feature vector for each of the songs. Although there might be better ways of capturing the harmony information, the use of these 7 dominant notes as elements of our feature vector did significantly aid the classification task.

Model Selection and Supervised Learning Results

At different stages of the project when different features were being tested, the mutual information metric was used to evaluate their usefulness. The KL-distance was used for computing the mutual information. While computing the KL distance is straightforward for the case of discrete feature vectors, the continuous feature vectors were dealt with by binning them and then using the discrete approach. The following figure (Fig. 3) lists the mutual information for each of the feature vectors considered.

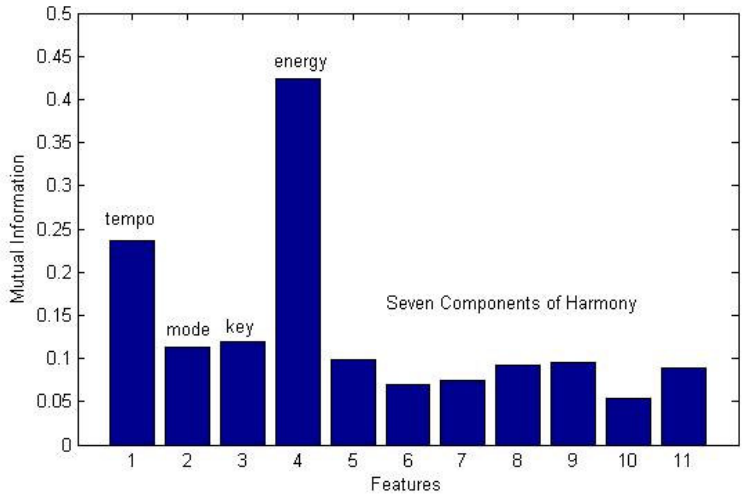

Figure 3 Mutual Information for different feature vectors

Having obtained a rough idea about the usefulness of the various features at hand, the forward search process was used to find the optimum set of features for classification through supervised learning using a Soft Margin SVM. The following table (Table 1) shows the progress at some of the steps in the forward search process. From the table, though it may seem that the feature vectors beyond energy and tempo didn't add much to the classification process, one must remember that marginal improvement of performance gets successively harder.

Feature Vector	SVM Kernel	Success Rate
Energy, Tempo	Linear	71.30%
Energy, Tempo, Mode	Linear	68.18%
Energy, Tempo, Key	RBF, σ = 3	69.70%
Energy, Tempo, Harmony	RBF, σ = 3	72.73%
Energy, Tempo, Harmony, Mode	RBF, σ = 3	75.76%

Table 1: Soft Margin SVM performance for some of the candidate feature sets and SVM kernels.

Depending on the set of feature vectors used, either linear or a Radial Basis Function (RBF) kernel seemed to give the best performance. In the case of simple features such as energy and tempo, where the relationship with mood is quite straightforward, a linear kernel performed best. The addition of harmony information introduced much more complexity to the feature space, and subsequently the RBF kernel gave the best results.

It was crucial for us to use the soft margin SVM because the training set was labeled manually. Since the perception of mood varies from person to person, there was a strong likelihood of some of the examples being labeled incorrectly. We varied the 'C' parameter to minimize the generalization error. In fact, the SVM module of Matlab that was used for classification scales the 'C' parameter for different training examples to account for the difference in the number of training examples for each of the classes.

Analysis

Having finalized the composition of our feature vector, choice of SVM Kernel etc., we performed k-fold cross validation in order to arrive at better estimates of the generalization error. We decided not to use k-fold cross validation for model selection since that would be computationally expensive and cumbersome. We also varied the size of the training set and averaged over the results of the iterations to obtain the following learning curve (Fig. 4).

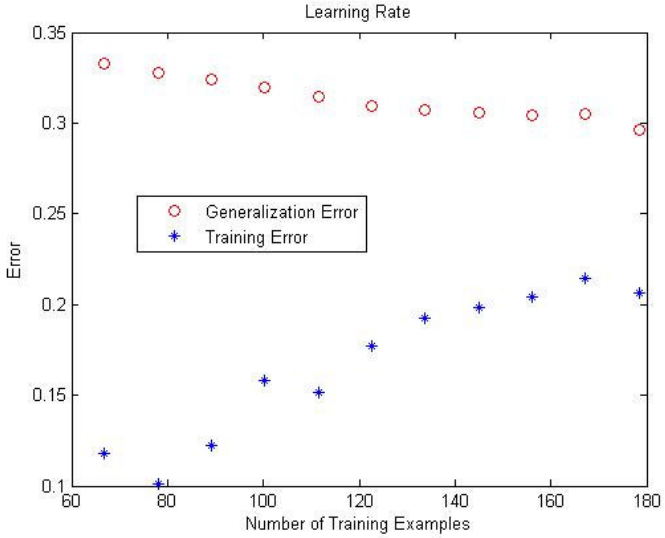

Figure 4 Learning Curve obtained through k-fold cross validation

The curve suggests that we are suffering from high variance. While we felt that with 157 training examples and a 10-dimensional feature vector we would be okay, it turns out that we are indeed over-fitting.

Unsupervised Learning

In order to gain more insight into our problem, we attempted unsupervised learning. If unsupervised learning worked well in clustering the dataset into Happy/Sad songs based on harmony alone, it would suggest that what we subjectively consider as being 'Happy' or 'Sad', correlates well with our harmony feature vector.

K-means clustering was run on the dataset with two clusters, harmony being the only feature vector. Based on the fact that the RBF gave best results for the features with harmony data, we hypothesized that K-means would not be able to do a great job clustering along the lines of happy and sad songs. However, we wanted to test it and see how well it could do.

As expected, if we assign labels to the clusters, the classification thus obtained was poor with an accuracy of 52.47%. In order to gain some visual understanding of why the clustering might be so difficult, we plotted the rank-2 approximation of the harmony feature data.

2-D Visualization of Harmony-only Feature Space

For visualization purposes, and as a sanity check on the data, we projected all of our 7-D harmony feature vectors into 2-D space. To project our higher dimensional data into 2-D, we computed the SVD (Singular Value Decomposition) of the Nx7 data matrix for each feature vector. We then selected the two eigenvectors of A^TA corresponding to the largest singular values of our data matrix, taken from the first two columns of the right singular matrix. We then projected each song's 7-D harmony feature vector onto the first and second

principal directions to obtain the coordinates of the feature vector in the 2-D space. Fig. 5 provides a good visualization for the high inseparability of the data, albeit visualized in 2-D. This helps to explain why K-means would do so poorly in separating the data. Further, it helps to verify why the RBF kernel worked best when harmony data was included in the feature vector, i.e. the RBF was able to carve out a complex decision surface for the best separation of the data.

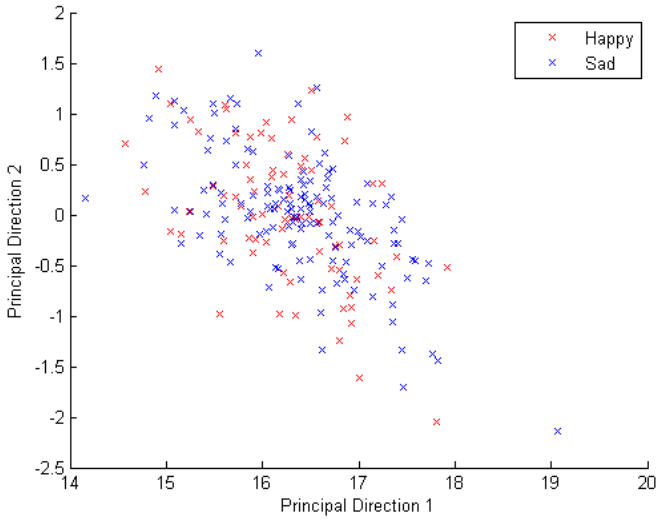

Figure 5: 2-D Low-Rank Approximation of 7-D Harmony Feature Data. Red points correspond to songs labeled 'Happy'. Blue points correspond to songs labeled 'Sad'.

Conclusion

The performance and capability of our algorithm can be significantly improved if we have access to a larger dataset because a larger dataset would allow us greater freedom in playing around with different ways of capturing the harmony information. Considering the subjective nature of mood classification, we believe that 70% success is a good result. The success of our algorithm is comparable to the results obtained by different research groups around the world. Papers in literature quote anywhere from 65% to 75% as the level of success achieved by their algorithms[1][2], though it should be noted that the classification results listed in the literature typically involve multi-class classification as opposed to our binary classification task.

References

[1] Cyril Laurier and Perfecto Herrera [2007], *Audio Music Mood Classification Using Support Vector Machine.* In Proceedings of the International Conference on Music Information Retrieval, Vienna, Austria.
[2] Lu, Liu and Zhang (2006), *Automatic Mood Detection and Tracking of Music Audio Signals*. IEEE Transactions on Audio, Speech, and Language Processing, Vol. 14, # 1, January 2006

Acknowledgements

We thank Prof. Andrew Ng, Andrew Maas and other members of the teaching staff for guiding us through the project. We also thank Abhishek Goel for helping us classify the list of 10,000 songs in our database. Finally, we thank Mayank Sanganeria for his valuable suggestions and help regarding feature selection.