Music Genre Classification

Xue Yu

Introduction

General information

Music genre classification is using machine learning automatically classfiy a song to its genre. Given a song, and then the song will be automatically classified as jazz, rock, pop, etc...

Music genre classification is a very interesting topic, it can be widely used, and itself is a very large topic, closely related with the the field of Music Information Retrieval.

Difficulties

Before I even sit down and try to code, I can think of two issues for this topic:

- what feature to use ?
- how to deal with multi genre songs ?

For a compressed mp3 song, it will take about 3 to 6 megabyte on hard disk, not even to mention a song may also contains lyrics and artist, how to transform a song to something computable, what feature to use is one issue.

Some songs belong to several genres, such as both rock and folk, It is hard to do this kind of classifications for computers.

Feature Extraction and Classifying

After read some research, I decide to use Mel-Frequency Cepstral Coefficients (MFCC)[1] as the feature of the song, and I would only classify one song to one genre.

Dataset

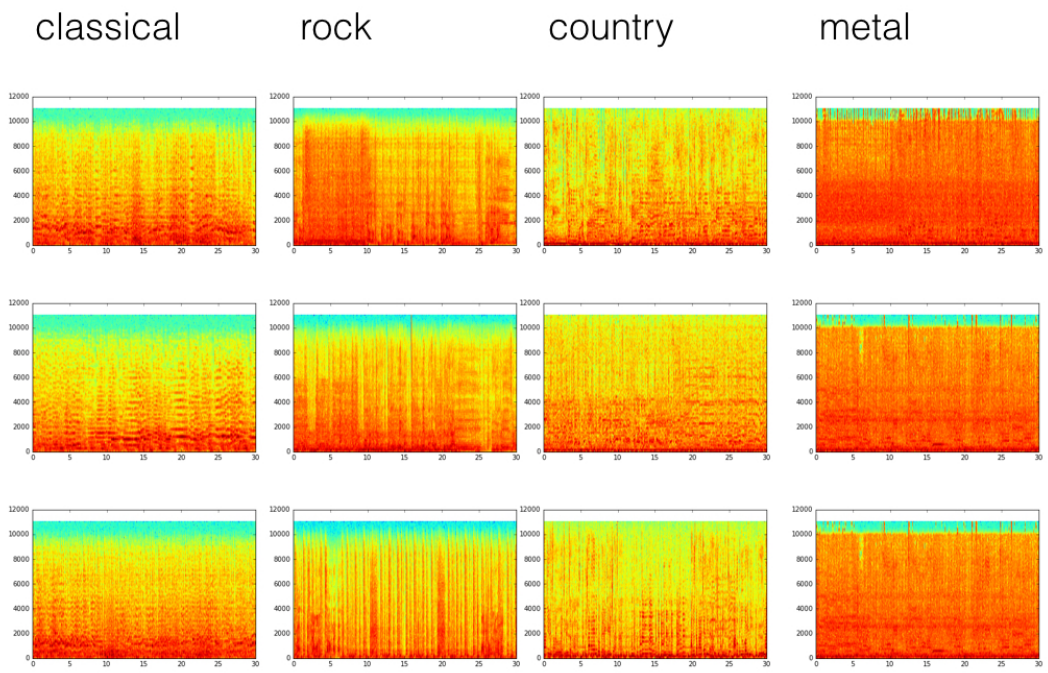
The dataset I use is the GTZAN Genre Collection[2]. This dataset contains 1000 songs, 10 genres, each kind 100 songs, I would only try to 4 genre: classical, country, metal, rock. 400 songs in total.

Approach

Preprocessing

The orignal dataset is in AU format, which is not easily used , so I use SOX[3] to transform the 400 songs to wav.

Then I randomly choose 3 songs of each kind and draw spectrum of them, one can tell by eyes that every genre has its characterics.



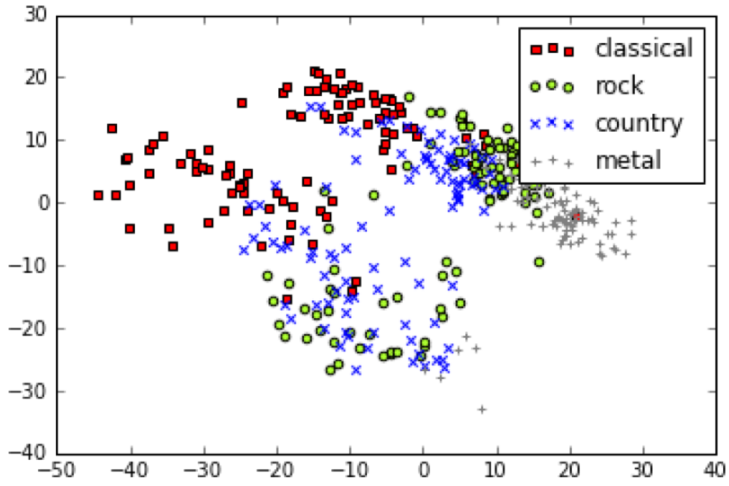
Then I use python_speech_features[4] to extract mfcc features of all songs.

For one song, it was transfered to about 3000 * 13 matrix, but 400 * 3000 * 13 still too much for me. So I calculate the mean of each column, thus I converted 400 songs to 400 * 13 numbers, Here I'm doing this with many doubts:

- Even I have proved that music is sepratable by just look at the specturm, but can mfcc feature be used, is it very genre related feature?
- Use one mean to reprensent about 3000 number, is this representive enough? Do I need to add variance and ranges , etc...
- The mfcc feature have some ranges, do I need to normalize these numbers?

Thus, I have transformed 400 songs to 400 * 13 numbers, or 400 * 14 numbers, if labeled added.

To see whether mfcc feature can be used here, I use Principal component analysis[5] to decrease 13 dimensions to 2, thus I can draw and get a visual feeling of how the music spread in space.



Actually from this we can see that classical and metal can be seprated by a linear classifier, and this is only the results of PCA, so mfcc feature is kind of realiable.

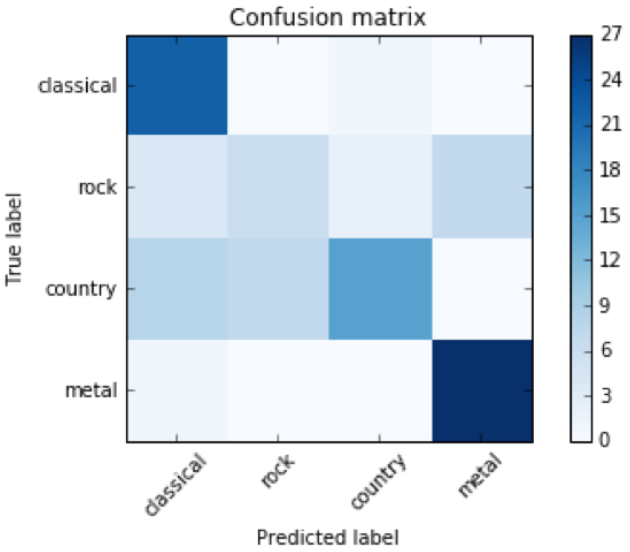
Till now, all my X and y are well prepared, now I can apply algothim on these numbers.

A Navie Classifier Approach

I split the data by 3:1, 75% as training data, 25% as test data. First I tried linear classifier with L2 regularization, it turns out it is actual learn, and the result is not bad.

On the training set and test set the accuracy is around 70%, this can prove that the classifier is learn, otherwise the accuracy would only around 25% since it is choosee 1 of 4.

From the confusion matrix, we can see almost no metal classified as classical and verse vise, just as the PCA told us : classical and metal can be well separted.



So Back to The questions I have raised:

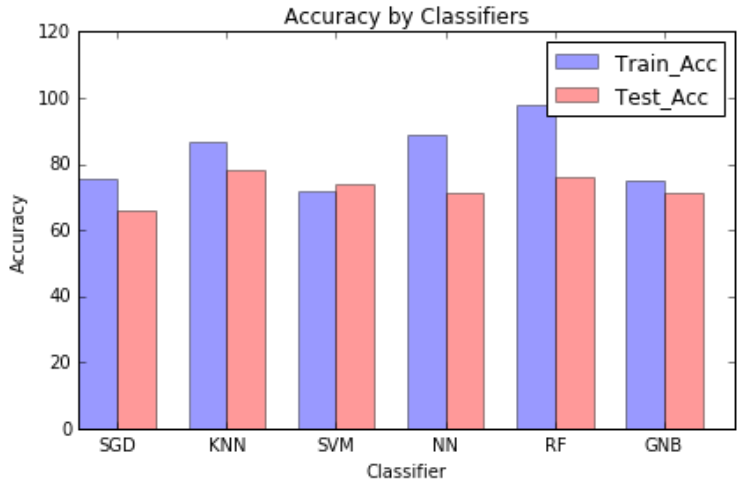
- The mfcc_mean can be used to represent song, and can be used to as the feature to classify gerne of song.
- I also tried to normalize the data, but the result doesn't improve much, So normalization actually not needed here since the numbers are not widely ranged, they fall between -30 to 30, so feature scaling not needed.

Other Classifier

Then I tried with different classifiers, the result is like this:

Classifier	Train_ACC	Test_ACC
SGDClassifier	75.32%	66.0%
k-Nearest Neighbors (K = 3)	86.67%	78.0%
Support Vector Machines(LinearSVC)	72.00%	74.0%
Neural Networks	86.67%	71.0%
Random Forests	98.00%	76.0%
Gaussian Naive Bayes	75.0%	71.0%

All the classifiers are just the standard classifiers from the Python sklearn pacakage, I didn't refine any parameters, I think they already did a good job in doing this.

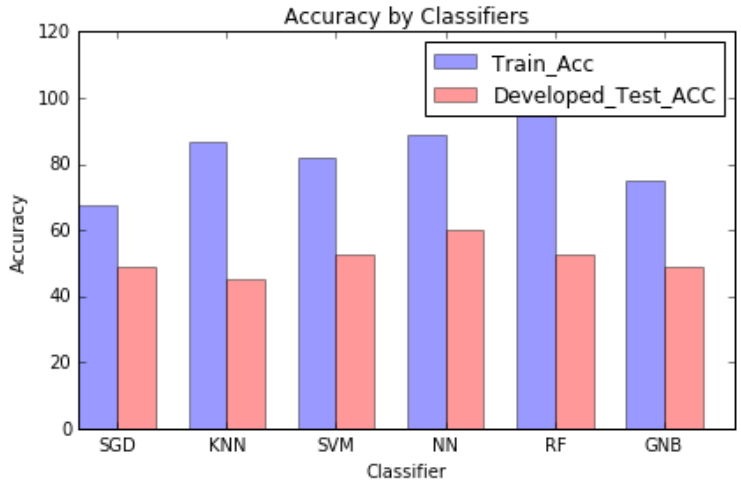


Real World Music

Then I download 20songs * 4 genres from Free Music Archive[6] , notice here first I have to transfer the music to the same format as the GTZAN dataset, that is : 30s 22050Hz Mono 16-bit

audio files in .wav format[7].

Classifier	Train_ACC	Test_ACC
SGDClassifier	67.33%	48.75%
k-Nearest Neighbors	86.67%	45.0%
Support Vector Machines	81.67%	52.5%
Neural Networks	89.0%	60.0%
Random Forests	99.0%	52.5%
Gaussian Naive Bayes	75.0%	48.75%



Not as high as the accuracy on test set, but apparent higher than random classify, from my point of view it already performed kind of well, because the music on the website is multi-genred.

Also notice that the train accuracy is different in the two histograms,because I use sklearn package to compute, due to the converge , the coefficients are slightly different every time.

Further Work

This may be applied to real world music, but need farther refinement, if lyrics and artist, more information are provided then I believe the classifier will behave more accuracy.

References

MIR_Feature_Extraction
:http://www.ifs.tuwien.ac.at/~schindler/lectures/MIR_Feature_Extraction.html

Building Machine Learning System with Python - Chapter 9

1. Mel-Frequency Cepstral Coefficients : https://en.wikipedia.org/wiki/Mel-frequency_cepstrum↩
2. GTZAN Dataset : http://marsyasweb.appspot.com/download/data_sets/↩
3. SOX : http://sox.sourceforge.net↩
4. python_speech_features : https://github.com/jameslyons/python_speech_features↩
5. Principal component analysis : https://en.wikipedia.org/wiki/Principal_component_analysis↩
6. Free Music Archive : http://freemusicarchive.org↩
7. iTunes: How to convert a song to a different file format : https://support.apple.com/en-us/HT204310↩