Hongyu Cao

Department of Mathematics

University of Washington

**ABSTRACT**

Different music genre have unique patterns so by listening lots of different music human can distinguish them. Computer can also accomplish this by transferring the audio file into something numerical, finding the patterns and treating them as a training set. Then give the computer test data and let it predict and calculate the accuracy.

**I. Introduction**

In this project, we have 3 cases and in every case we need to classify the music genres. The difference of these case is only the input. I used Naïve Bayes method for all of them and after calculating the accuracy I also compared the impact of choosing different features.

**II. Theoretical Background**

In order to convert the audio file into something numerical, we need to introduce the concept spectrogram. This function uses short-time Fourier transform and each column of its result contains an estimate of the short-term, time-localized frequency content of the input.

By using SVD we can capture the coefficient of the most important feature of the spectrogram.

It’s a supervised method since we’ve knew how many genres we want to classify.

**III. Algorithm Implementation and Development**

It totally has 3 cases and all of them we need to start from building training set.

Case 1: We need to consider 3 different bands of different genres so I chose Beethoven symphony no. 1 and 3, Taylor Swift 1989 and Miles Davis Kind of Blue. I choose 10 evenly separate points of each piece as my 5-second clip starting points. But the number of songs of each genre is different. Totally, we have 80 clips for Beethoven, 130 for Taylor and 60 for Miles. After gather all data, I did a SVD on it. We randomly select 40 clips of each genre as training set and use the left as test data. I repeated this method for 200 times and calculated the accuracy in percentage.

Case 2: We need to consider 3 different bands the same genre but from different composers so I choose many Beethoven sonatas, Mozart sonatas and many pieces from Chopin. Chopin’s pieces are more diverse and they include his Waltz, Prelude, Sonata, Nocturne etc. Do a SVD for the collected data and choose 200 out of 260 clips as Beethoven training set, 80 out of 120 as Mozart training set and 400 out of 520 as Chopin training set.

Case 3: I used the audio file from the link in the canvas discussion board and I picked classical, jazz, and rock as 3 genres. Each genre has 100 pieces and I choose 3 evenly separated points as starting points so totally we have 900 5-second clips. In classifying step, I used Naïve Bayes method to train those data and do a cross validation.

**Sec. IV. Computational Results**

Case 1:

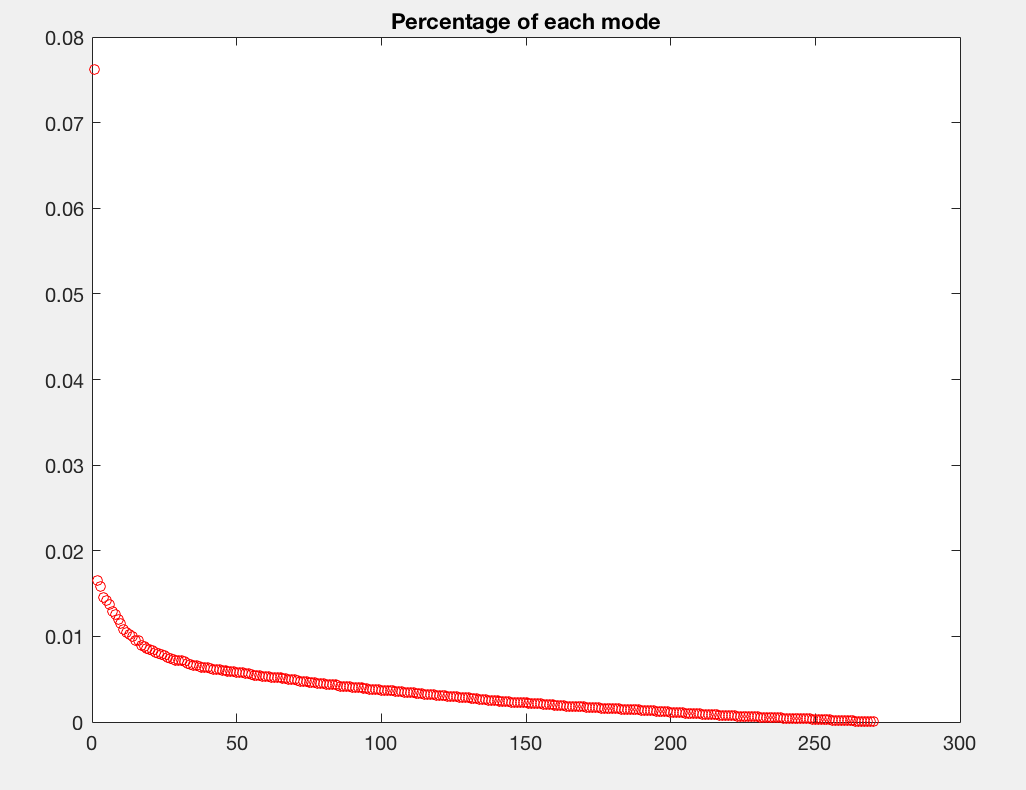


Figure 1: We want to the feature that can distinguish each clips so we should skip the first one.

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Genre/Accuracy | 2:4 | 2:10 | 2:50 | 2:100 |
| Classical | 0.7656 | 0.8191 | 0.8161 | 0.8329 |
| Country | 0.7771 | 0.8487 | 0.8317 | 0.8321 |
| Jazz | 0.5571 | 0.6009 | 0.6344 | 0.6164 |

Table 1: Cross validation result

Case 2:

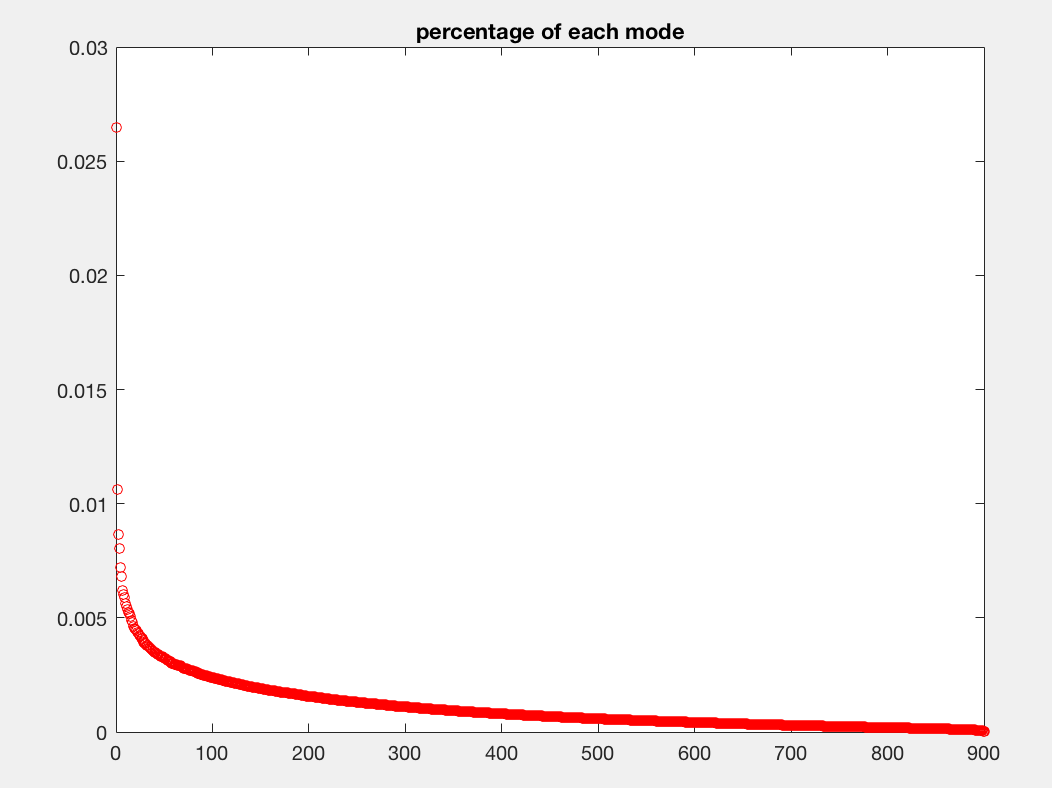


Figure 2: For this case, the first mode is also much bigger than others.

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Genre/Accuracy | 2:4 | 2:10 | 2:50 | 2:100 |
| Beethoven | 0.7641 | 0.6479 | 0.2935 | 0.2554 |
| Mozart | 0.5062 | 0.7971 | 0.8056 | 0.7986 |
| Chopin | 0.6833 | 0.3833 | 0.3017 | 0.2878 |

Table 2: cross validation for case 2

Case 3:

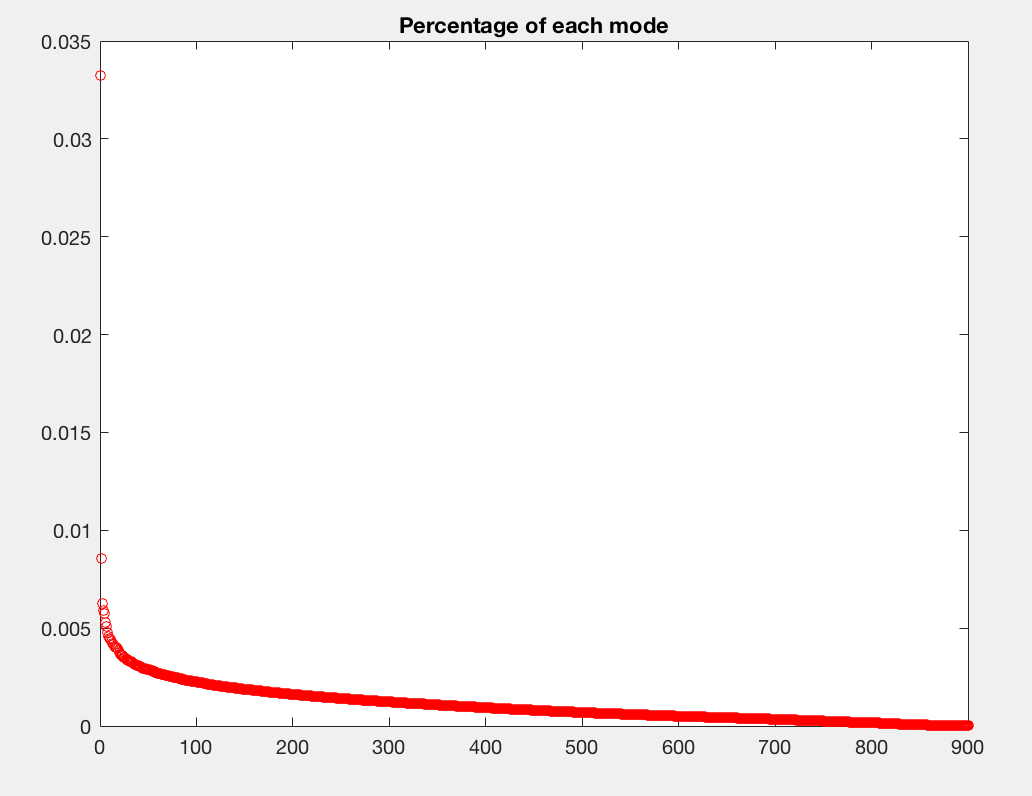


Figure 3: The plot shows each percentage of each mode.

Accuracy: Take 200 clips from each genre as training set and the left 100 clips as test set. Repeat this method for 200 times to get an average accuracy. Also we need to try different feature by changing the selected V.

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Genre/Accuracy | 2:4 | 2:10 | 2:50 | 2:100 |
| Classical | 0.8952 | 0.9069 | 0.9002 | 0.8928 |
| Jazz | 0.3865 | 0.3263 | 0.2702 | 0.1817 |
| Rock | 0.6715 | 0.6026 | 0.4838 | 0.4400 |

**Sec. V. Summary and Conclusions**

For case one, the overall accuracy is not bad. Especially classical and country ones did very well. But Jazz’s accuracy is low due to the number of samples is less than the other.

Case 2, by choosing V from 2rd to 4th column, all of their accuracy is good. Also the original data could also have an impact on it because all of the pieces of Mozart and Beethoven that I chose are piano sonatas but Chopin’s works are more diverse and he only have 2 piano sonatas. As the number of columns of V increase, the accuracy of Chopin decreased fast.

In Case 3, Jazz has an obvious low accuracy than others though I don’t know why. But for case 3, the number of sample clips for each genre is the same.

**Appendix A MATLAB functions used and brief implementation explanation**

s = spectrogram(x) returns the short-time Fourier transform of the input signal, x. Each column of s contains an estimate of the short-term, time-localized frequency content of x.

[y,Fs] = audioread(filename) reads data from the file named filename, and returns sampled data, y, and a sample rate for that data, Fs.

**Appendix B MATLAB codes**

The code is case 3 and the other 2 cases are similar.

%% Read music clips

list1=dir('/Users/hongyu/Downloads/genres/classical/\*');

list2=dir('/Users/hongyu/Downloads/genres/jazz/\*');

list3=dir('/Users/hongyu/Downloads/genres/rock/\*');

l=[list1;list2;list3];

%%

list=[];

for i = 1: 306

if ~strcmp(l(i).name,'.') && ~strcmp(l(i).name,'..')

list=[list;l(i)];

end

end

%% Pick 3 5-seoncd clips of each piece

s=[];

l=list;

for i=1:300

[y, fs]=audioread((strcat(l(i).folder,'/',l(i).name)));

[m,n]=size(y);

interval=m/3;

start=[1];

for j=1:3

start=[start,floor(interval\*j)];

y1 = y(start(j):start(j)+5\*fs);

A=spectrogram(y1);

A=reshape(A,[],1);

s=[s,A];

end

end

%% SVD

s=abs(s);

[U,S,V]=svd(s,'econ');

%%

plot(diag(S)/sum(diag(S)),'ro','Linewidth',[0.5],'MarkerSize', 5),title('Percentage of each mode');

%% corss validation and calculate accuracy

a=0;

b=0;

c=0;

for t = 1:200

q1=randperm(300);

q2=randperm(300);

q3=randperm(300);

xB=V(1:300,2:50);

xT=V(301:600,2:50);

xM=V(601:900,2:50);

n=200;

xtrain=[xB(q1(1:n),:);xT(q2(1:n),:); xM(q3(1:n),:)];

xtest=[xB(q1(n+1:end),:);xT(q2(n+1:end),:); xM(q3(n+1:end),:)];

ctrain=[ones(n,1);2\*ones(n,1);3\*ones(n,1)];

nb=fitcnb(xtrain,ctrain);

pre=nb.predict(xtest);

%bar(pre)

for i = 1: 300

if i<101 && pre(i)==1

a=a+1;

else if i>100 && i<201 && pre(i)==2

b=b+1;

else if i>200 && pre(i)==3

c=c+1;

end

end

end

end

end

a=a/20000;

b=b/20000;

c=c/20000;