Music Genre Identification

Author: Seoyoung Park

Abstract

There are a number of genres in music, and when we listen to a song, we can approximately guess what kind of genre the song is. This implies that there is a way to classify different genres with what we are familiar of. In this paper, we are going to classify different bands and genres by singular value decomposition, using a concept of clustering and classification.

Introduction and Overview

The main purpose of this paper is to explore clustering and classification with different data sets. Taking 5-second clips from several songs of several artists in several genres, we can make a data set to do the singular value decomposition, try to classify the data and perform cross validation. In this paper, three tests are going to be shown. For the first test, we are going to consider three different bands of different genres. For the second, the test 1 is repeated but the bands are chosen within same genre. Finally for the last test, 3 different genres are going to be tested with several bands within each genre.

Theoretical Background

Clustering is a type of unsupervised machine learning which aims to find homogeneous subgroups such that clusters are more similar to each other than the others. For data mining, we want to make predictions based upon the given data and apply it to new data. There are two basic learnings for this: one is unsupervised learning and the other is supervised learning. Unsupervised learning is a type of machine learning algorithm used to draw inferences from datasets consisting of input data without labeled responses. The goal of the unsupervised learning is to look for patterns of the data and discover clustering in some kind of space. Based on what you already know, supervised learning would be able to classify data. The idea of supervised learning is labeling data ahead of time so that you can make future decisions. Also, in supervised learning, it is trying to do the classification and regressions. If we can do the classification, we can also find basically some kind of the best least square fit curve to the data.

One of classifications, which is used in this paper, is Naive Bayes classification. Basically, it assumes that the presence of a particular feature in a class is unrelated to the presence of any other feature. It can be expressed as following

$$P(c|x) = \frac{P(x|c)P(c)}{P(x)}$$
$$P(c|x) = P(x_1|c) * P(x_2|c) * ... * P(x_n|c) * P(c)$$

where $P(c|x)$ is the posterior probability of test(c) given training set(x), $P(c)$ is the prior probability of test(c), $P(x|c)$ is the likelihood which is the probability of training set(x) given test(c), and $P(x)$ is the prior probability of training set(x).

Algorithm Implementation and Development

1. First, the function 'audioConvert' was made to convert the audio files(mp3), which are selected to be tested, to the spectrogram of a cropped 5-second song sample. Do this progress for each category such as each band or each genre. I took 10 samples from each band for test 1 and 2(overall 30 samples for each test) and 20 samples from each genre for test 1(60 samples overall), and remember each sample is 5-seconds long.
2. After collecting all samples, combine all data into the one data matrix X so that each column of the matrix X is corresponding to each clip and do the singular value decomposition to get matrix U,S,and V.
3. From the matrix V resulted from SVD so it contains information regarding the spectrogram's projections onto the principal components, divide the matrix V into three so each matrix corresponds to each artist or genre.
4. To cross validate, 80% of the data is taken for the training set and the remaining 20% of the data is taken for the test, assigning random permutations.
5. Use Naive Bayes classification and test 100 times, calculating overall correctness.

Computational Results

Test 1
The goal of the test 1 is to classify three different bands from three different genres. Adele, Eminem, and Bob Marley were chosen to represent R&B, Hip Hop, and Reggae respectively. For each band, 5 songs were chosen, and for each song, 10 of 5-second samples were selected, so there are total of 50 samples for each band.

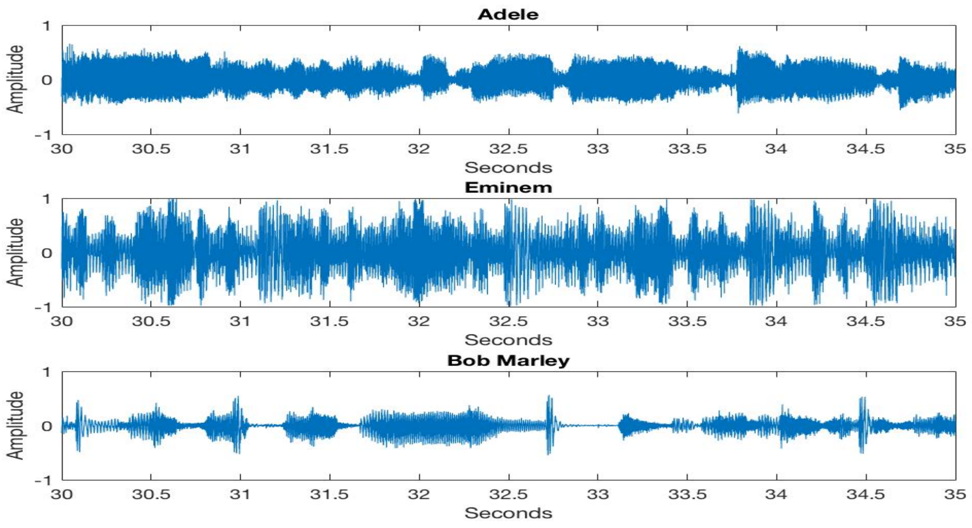


Figure 1: Three pieces of songs of three different bands from different genres are demonstrated. 5-second samples(part from 30sec-35sec) of Adele - Someone Like You(R&B), Eminem - The Way I Am(Hip Hop), and Bob Marley - Roots Rock Reggae(Reggae) are given.

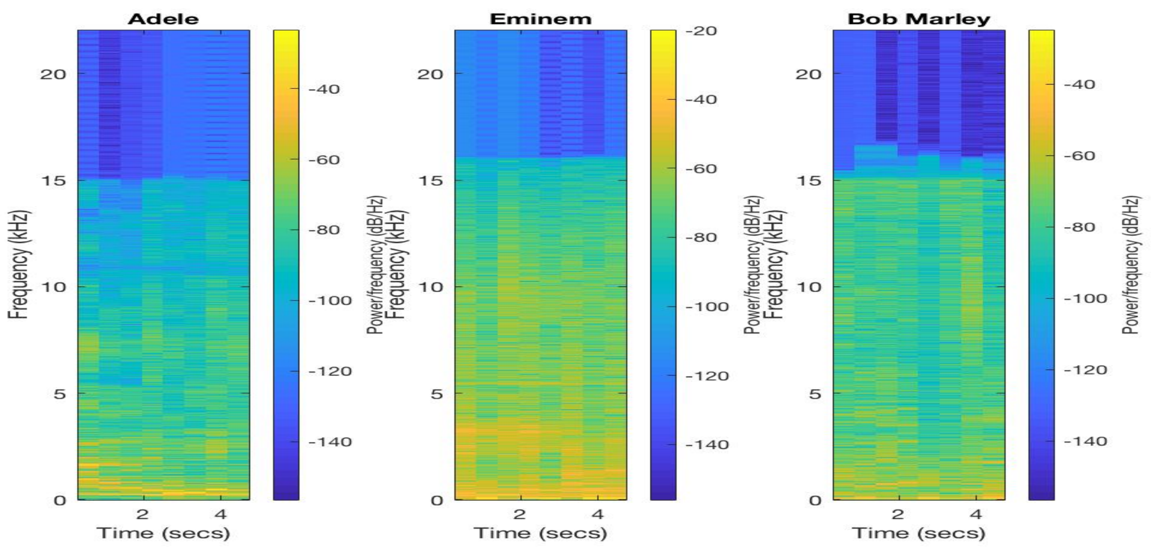
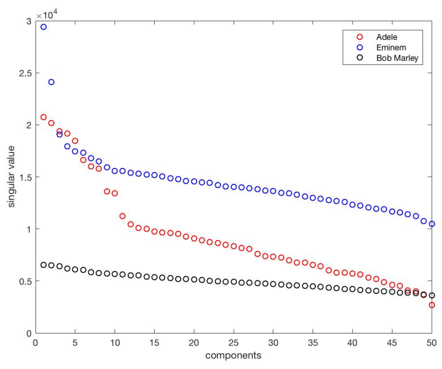
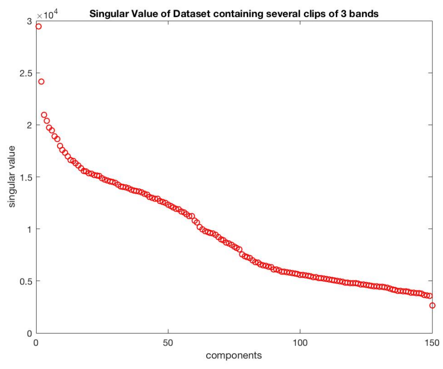


Figure 2: Shows spectrograms of each band. Spectrogram can hold both time and frequency information of the sound signal.

Since the bands from 3 different genres were chosen to be tested, spectrogram and frequency of different genres would look quite different, and figure 1 and 2 are examples of frequency and spectrogram of one song of each band to represent how each genre approximately looks like. Also, the overall singular values for each band are quite different as shown in figure 3(a).



(a) The dataset of each band



(b) The dataset of all 3 bands

Figure 3: Shows singular value of each component. Since the dataset is given by song samples of three different bands from different genres, the singular value of three bands look varied a lot.

Table 1: Different Genres

Band	Accuracy(%)
Adele	65.4%
Eminem	92.9%
Bob Marley	48.7%
Overall	69%

80% of the samples were chosen to be training set and the remainder were to be the test, so 40 samples of each band are in the training set, and 10 samples are tested to see if it's classified well using Naive Bayes classification. Then, the test were repeated 100 times to get the average accuracy. If you see the figure 3(a), singular values for Eminem are way apart from the others, so it got the highest accuracy. And since singular values of Adele and Bob Marley are closer than those of Eminem, the percentages of accuracy for Adele and Bob Marley are lower than Eminem's. But still, overall, it was classified pretty well because different bands from different genres were tested.

Test 2

The goal of the test 2 is to classify three different bands within same genre and compare the result to the test 1. Drake, Rae Sremmurd, Lil Wayne were chosen to represent Hip Hop. The same process of test 1 was repeated.

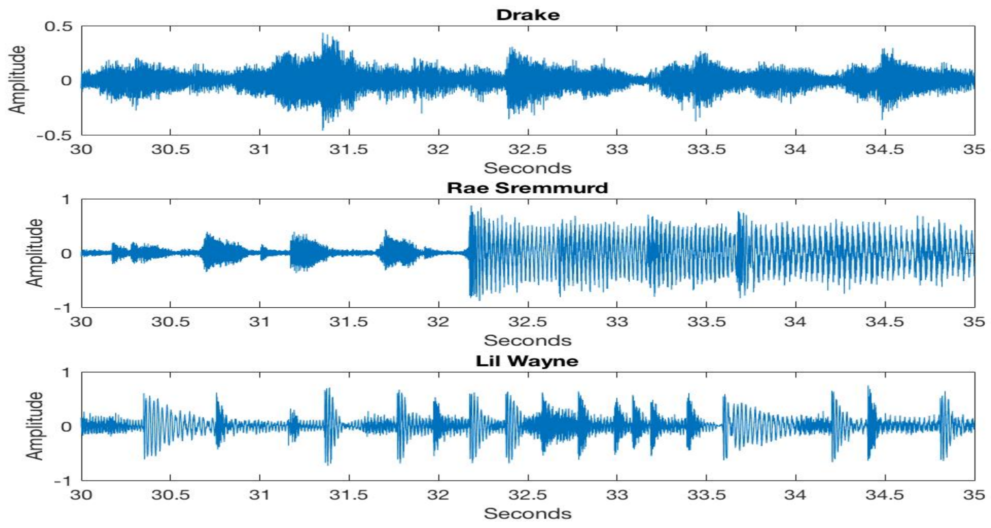


Figure 4: Three pieces of songs of three different bands within same genre are demonstrated. 5-second samples(part from 30sec-35sec) of Drake - We Made It, Rae Sremmurd - Look Alive, and Lil Wayne - Lollipop are given.

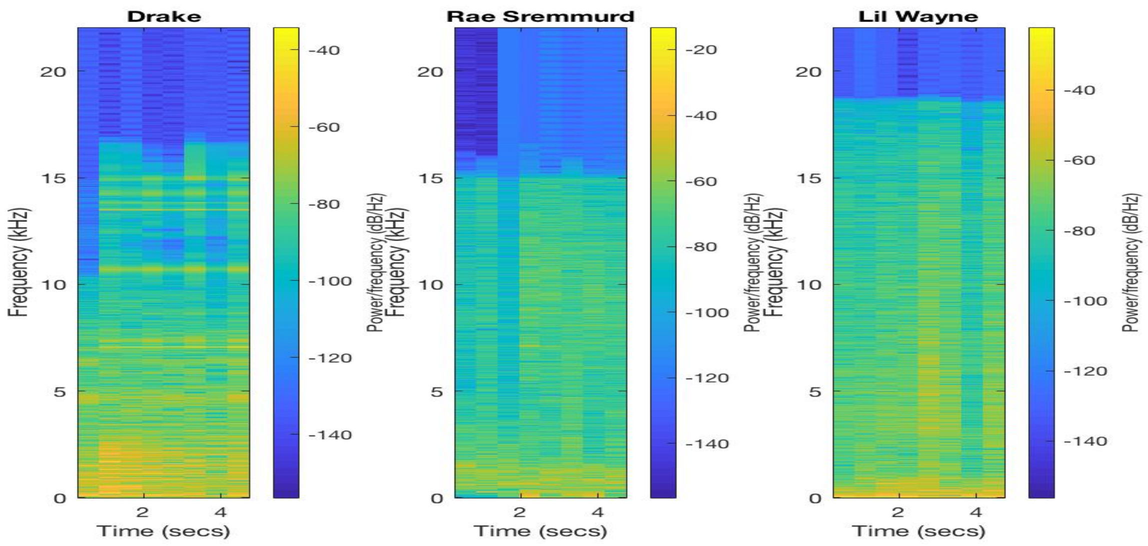
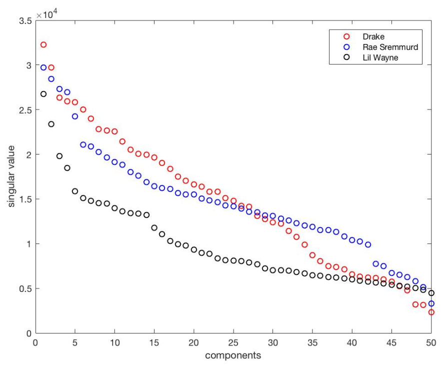
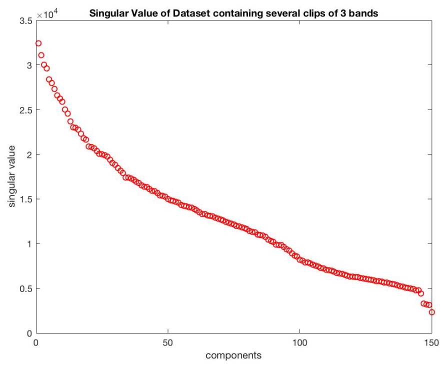


Figure 5: Shows spectrograms of each band. Spectrogram can hold both time and frequency information of the sound signal.



(a) The dataset of each band



(b) The dataset of all 3 bands

Figure 6: Shows singular value of each component. Since the dataset is given by song samples of three different bands within same genre, the singular value of three bands look similar.

Table 2: Same Genre

Band	Accuracy(%)
Drake	24.8%
Rae Sremmurd	75.5%
Lil Wayne	56.3%
Overall	52.2%

Even though 3 different bands were chosen, because they are all in the same genre, overall singular values of each band are similar(figure 6(a)) compared to the test 1(figure 3(a)). Notice that the overall accuracy is less than that of test 1 because all three bands share similar features. Therefore, the classification was more difficult for the test 2 than test 1.

Test 3

The goal of the test 3 is to classify three different genres, and several bands are selected within each genre. In this paper, Jazz, Classical, and Pop songs were chosen to be tested. For each genre, 10 song were chosen, and for each song, 10 of 5-second samples were selected, so there are total of 100 samples for each genre.

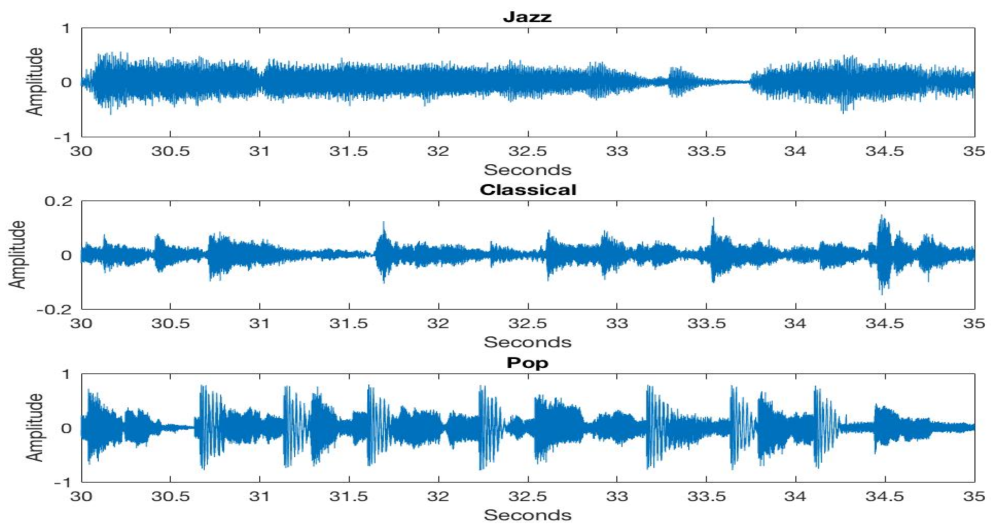


Figure 7: Three different genres are demonstrated. 5-second samples(30sec-35sec) of Miles Davis - Kind of Blue(Jazz), Vivaldi - Andante(Classical), and Taylor Swift - Blank Space(Pop) are given.

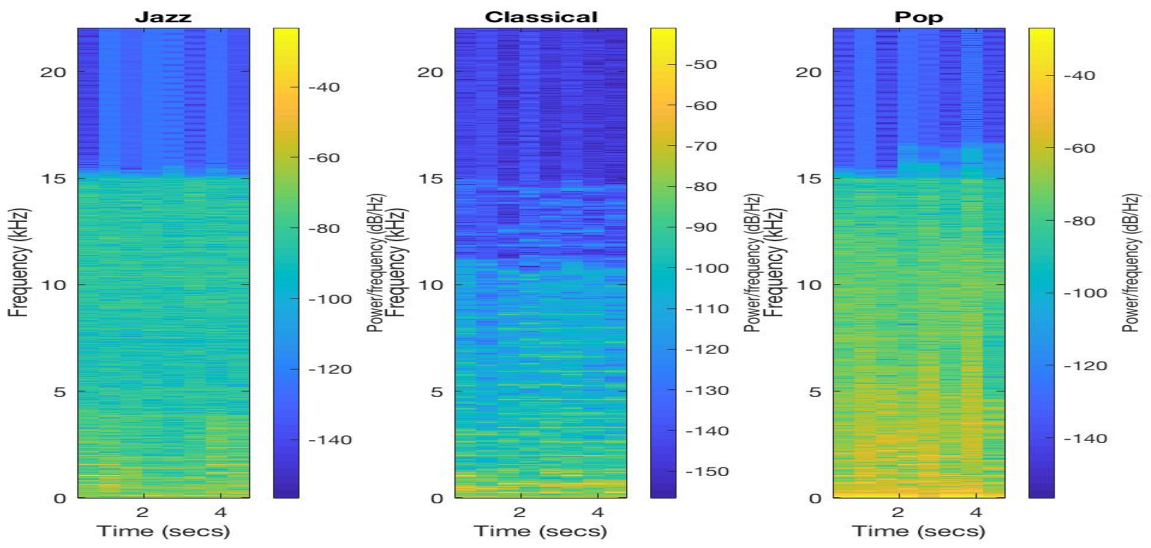
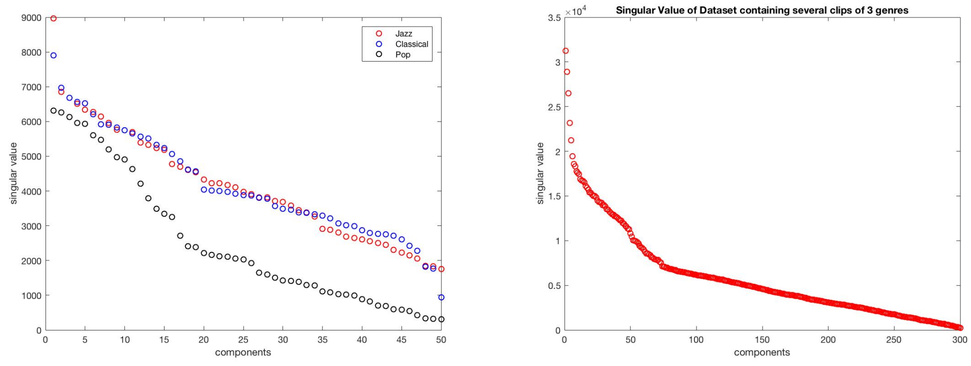


Figure 8: Shows spectrograms of each band. Spectrogram can hold both time and frequency information of the sound signal.



(a) The dataset of each genre (b) The dataset of all 3 genres

Figure 9: Shows singular value of each component.

Table 3: Genre Classification

Genre	Accuracy(%)
Jazz	47.8%
Classical	31.2%
Pop	90.45%
Overall	56.48%

80 samples of each genre are in the training set, and 20 samples are tested to see if it's classified well using Naive Bayes classification for 100 times. Notice that since each genre has fairly distinct musical features, the classification performed well enough(56.48%). If you see figure9 (a), singular values for Jazz and Classical are similar while Pop is away from those two. Therefore, accuracy percentages for Jazz and Classical result in less accuracy than that for Pop.

Summary and Conclusions

Band and genre classification were tested with spectrograms of 5-second samples of songs. Three different bands from three different genres were tested in the test 1, three different bands from the same genre were tested in the test 2, and lastly, three different genres were tested in the test 3. Combining all samples, Singular Value Decomposition was performed under the data matrix. With the resultant V matrix, Naive Bayes Classification was tested, then the percentage of accuracy was calculated. The classification worked well on test 1 and test 2, but test 3 was poorly classified due to samples sharing similar music features within same genre.

Appendix A

[Y, FS] = audioread(FILENAME): reads an audio file specified by the string FILE, returning the sampled data in Y and the sample rate FS, in Hertz.
S = spectrogram(X): returns the short-time Fourier transform of the signal specified by vector X in the matrix S.
real(X): is the real part of X.
F = fullfile(FOLDERNAME1, FOLDERNAME2, ..., FILENAME): builds a full file specification F from the folders and file name specified.
dir NAME: lists the files in a folder.
P = randperm(N): returns a vector containing a random permutation of the integers 1:N.
MODEL=fitcnb(TBL,Y): returns a naive Bayes model MODEL for data in the table TBL and response Y. TBL contains the predictor variables.

Appendix B

```
function [reshaped, signal] = audioConvert(t, filename)
[signal,fs] = audioread(filename);
signal = (signal(:,1)+ signal(:,2))/2;
t1 = t*fs;
t2 = (t+5)*fs;
crop = signal(t1:t2);
sample_real = real(spectrogram(crop,fs));
reshaped = sample_real(:);
end


clear all; close all; clc;

L=5; n=220501;
k=(2*pi/L)*[0:n/2-1 -n/2:-1]; ks=fftshift(k);

[y1,fs1] = audioread('Someone Like You.wav');
y1 = (y1(:,1)+y1(:,2))/2;
dt1 = 1/fs1;
cropped1 = y1(30*fs1:35*fs1);
figure(1); subplot(3,1,1); grid on; plot(linspace(30,35,length(cropped1)),cropped1)
title('Adele');xlabel('Seconds'); ylabel('Amplitude');
%figure(2); plot(psd(spectrum.periodogram,y,'Fs',fs,'NFFT',length(cropped)));
figure(2); subplot(1,3,1); spectrogram(cropped1,[],[],[],fs1,'yaxis'); title('Adele')

[y2,fs2] = audioread('The Way I Am.wav');
y2 = (y2(:,1)+y2(:,2))/2;
dt2 = 1/fs2;
```

```
cropped2 = y2(30*fs2:35*fs2);
figure(1); subplot(3,1,2); grid on; plot(linspace(30,35,length(cropped2)),cropped2)
title('Eminem');xlabel('Seconds'); ylabel('Amplitude');
%figure(2); plot(psd(spectrum.periodogram,y,'Fs',fs,'NFFT',length(cropped)));
figure(2); subplot(1,3,2); spectrogram(cropped2,[],[],[],fs2,'yaxis'); title('Eminem')

[y3,fs3] = audioread('Reggae.wav');
y3 = (y3(:,1)+y3(:,2))/2;
dt2 = 1/fs3;
cropped3 = y3(30*fs3:35*fs3);
figure(1); subplot(3,1,3); grid on; plot(linspace(30,35,length(cropped3)),cropped3)
title('Bob Marley');xlabel('Seconds'); ylabel('Amplitude');
%figure(2); plot(psd(spectrum.periodogram,y,'Fs',fs,'NFFT',length(cropped)));
figure(2); subplot(1,3,3); spectrogram(cropped3,[],[],[],fs3,'yaxis'); title('Bob Marley')

% figure;
% subplot(3,1,1), plot(linspace(30,35,length(cropped1)),cropped1,'k')
% subplot(3,1,2), plot(linspace(30,35,length(cropped1)),fft(cropped1),'k')
% subplot(3,1,3), plot(ks(1,n/2:n),abs(fftshift(cropped1))/max(abs(cropped1))) %drawnow


clear all; close all; clc;

% Adele
AD = cell(1,50);
myFolder = '/Users/emilypark/Desktop/WI2018/AMATH482/HW3/test1/Adele';
filePattern = fullfile(myFolder, '*.mp3');
theFiles = dir(filePattern);
for i=1:5
    baseFileName = theFiles(i).name;
    fullFileName = fullfile(myFolder, baseFileName);
    AD{i} = audioConvert(30, fullFileName);
    AD{i+5} = audioConvert(45, fullFileName);
    AD{i+10} = audioConvert(50, fullFileName);
    AD{i+15} = audioConvert(60, fullFileName);
    AD{i+20} = audioConvert(70, fullFileName);
    AD{i+25} = audioConvert(80, fullFileName);
    AD{i+30} = audioConvert(100, fullFileName);
    AD{i+35} = audioConvert(110, fullFileName);
    AD{i+40} = audioConvert(150, fullFileName);
    AD{i+45} = audioConvert(160, fullFileName);
end

% Eminem
EM = cell(1,50);
myFolder = '/Users/emilypark/Desktop/WI2018/AMATH482/HW3/test1/Eminem';
filePattern = fullfile(myFolder, '*.mp3');
theFiles = dir(filePattern);
for i=1:5
    baseFileName = theFiles(i).name;
    fullFileName = fullfile(myFolder, baseFileName);
    EM{i} = audioConvert(30, fullFileName);
    EM{i+5} = audioConvert(45, fullFileName);
    EM{i+10} = audioConvert(50, fullFileName);
    EM{i+15} = audioConvert(60, fullFileName);
    EM{i+20} = audioConvert(70, fullFileName);
    EM{i+25} = audioConvert(80, fullFileName);
    EM{i+30} = audioConvert(100, fullFileName);
    EM{i+35} = audioConvert(110, fullFileName);
    EM{i+40} = audioConvert(150, fullFileName);
    EM{i+45} = audioConvert(160, fullFileName);
end

% Bob Marley
BM = cell(1,50);
myFolder = '/Users/emilypark/Desktop/WI2018/AMATH482/HW3/test1/Bob Marley';
filePattern = fullfile(myFolder, '*.mp3');
theFiles = dir(filePattern);
for i=1:5
    baseFileName = theFiles(i).name;
    fullFileName = fullfile(myFolder, baseFileName);
    BM{i} = audioConvert(30, fullFileName);
    BM{i+5} = audioConvert(45, fullFileName);
    BM{i+10} = audioConvert(50, fullFileName);
    BM{i+15} = audioConvert(60, fullFileName);
    BM{i+20} = audioConvert(70, fullFileName);
    BM{i+25} = audioConvert(80, fullFileName);
    BM{i+30} = audioConvert(100, fullFileName);
    BM{i+35} = audioConvert(110, fullFileName);
    BM{i+40} = audioConvert(150, fullFileName);
    BM{i+45} = audioConvert(160, fullFileName);
end

X = zeros(3*50,294921);
for i=1:50
    X(i,:) = AD{i}(:,1);
    X(i+(50),:) = EM{i}(:,1);
    X(i+(100),:) = BM{i}(:,1);
end
x1 = X(1:50,:);
x2 = X(51:100,:);
x3 = X(101:150,:);
[u1,s1,v1] = svd(x1,'econ');
figure(1);
plot(diag(s1),'ro')
[u2,s2,v2] = svd(x2,'econ');
hold on;
plot(diag(s2),'bo')
[u3,s3,v3] = svd(x3,'econ');
plot(diag(s3),'ko')
xlabel('components'); ylabel('singular value'); legend('Adele','Eminem','Bob Marley');

[m,n] = size(X);
[u,s,v] = svd(X,'econ');
%Y = u.' * X;
%Cy=(1/(n-1))*(Y)*(Y.');
%PCs = diag(Cy);
figure(2);plot(diag(s),'ro')
xlabel('components'); ylabel('singular value'); title('Singular Value of Dataset containing severa


ad = v(:,1:50);
em = v(:,51:100);
bm = v(:,101:150);

o_ad = 0;
o_em = 0;
o_bm = 0;
```

```
for i = 1:100
q1 = randperm(50); q2 = randperm(50); q3 = randperm(50);
train = [ad(q1(1:40),:); em(q2(1:40),:); bm(q3(1:40),:)];
test = [ad(q1(41:end),:); em(q2(41:end),:); bm(q3(41:end),:)];
train2 = [ones(40,1);2*ones(40,1);3*ones(40,1)];
ox = [ones(10,1);2*ones(10,1);3*ones(10,1)];

Mdl = fitcnb(train,train2);
prediction = Mdl.predict(test);

for j = 1:30
    if prediction(j)-ox(j) == 0
        if j <= 10
            o_ad = o_ad + 1;
        elseif j <= 20
            o_em = o_em + 1;
        else
            o_bm = o_bm + 1;
        end
    end
end
end


clear all; close all; clc;

L=5; n=220501;
k=(2*pi/L)*[0:n/2-1 -n/2:-1]; ks=fftshift(k);

[y1,fs1] = audioread('We Made It.wav');
y1 = (y1(:,1)+y1(:,2))/2;
dt1 = 1/fs1;
cropped1 = y1(30*fs1:35*fs1);
figure(1); subplot(3,1,1); grid on; plot(linspace(30,35,length(cropped1)),cropped1)
title('Drake');xlabel('Seconds'); ylabel('Amplitude');
%figure(2); plot(psd(spectrum.periodogram,y,'Fs',fs,'NFFT',length(cropped)));
figure(2); subplot(1,3,1); spectrogram(cropped1,[],[],[],fs1,'yaxis'); title('Drake')

[y2,fs2] = audioread('Look Alive.wav');
y2 = (y2(:,1)+y2(:,2))/2;
dt2 = 1/fs2;
cropped2 = y2(30*fs2:35*fs2);
figure(1); subplot(3,1,2); grid on; plot(linspace(30,35,length(cropped2)),cropped2)
title('Rae Sremmurd');xlabel('Seconds'); ylabel('Amplitude');
%figure(2); plot(psd(spectrum.periodogram,y,'Fs',fs,'NFFT',length(cropped)));
figure(2); subplot(1,3,2); spectrogram(cropped2,[],[],[],fs2,'yaxis'); title('Rae Sremmurd')

[y3,fs3] = audioread('Lollipop.wav');
y3 = (y3(:,1)+y3(:,2))/2;
dt2 = 1/fs3;
cropped3 = y3(30*fs3:35*fs3);
figure(1); subplot(3,1,3); grid on; plot(linspace(30,35,length(cropped3)),cropped3)
title('Lil Wayne');xlabel('Seconds'); ylabel('Amplitude');
%figure(2); plot(psd(spectrum.periodogram,y,'Fs',fs,'NFFT',length(cropped)));
figure(2); subplot(1,3,3); spectrogram(cropped3,[],[],[],fs3,'yaxis'); title('Lil Wayne')

% figure;
% subplot(3,1,1), plot(linspace(30,35,length(cropped1)),cropped1,'k')
% subplot(3,1,2), plot(linspace(30,35,length(cropped1)),fft(cropped1),'k')
% subplot(3,1,3), plot(ks(1,n/2:n),abs(fftshift(cropped1))/max(abs(cropped1))) %drawnow


clear all; close all; clc;

% Drake
DR = cell(1,50);
myFolder = '/Users/emilypark/Desktop/WI2018/AMATH482/HW3/test2/Drake';
filePattern = fullfile(myFolder, '*.mp3');
theFiles = dir(filePattern);
for i=1:5
    baseFileName = theFiles(i).name;
    fullFileName = fullfile(myFolder, baseFileName);
    DR{i} = audioConvert(30, fullFileName);
    DR{i+5} = audioConvert(45, fullFileName);
    DR{i+10} = audioConvert(50, fullFileName);
    DR{i+15} = audioConvert(60, fullFileName);
    DR{i+20} = audioConvert(70, fullFileName);
    DR{i+25} = audioConvert(80, fullFileName);
    DR{i+30} = audioConvert(100, fullFileName);
    DR{i+35} = audioConvert(110, fullFileName);
    DR{i+40} = audioConvert(150, fullFileName);
    DR{i+45} = audioConvert(160, fullFileName);
end

% Rae Sremmurd
RS = cell(1,50);
myFolder = '/Users/emilypark/Desktop/WI2018/AMATH482/HW3/test2/Rae Sremmurd';
filePattern = fullfile(myFolder, '*.mp3');
theFiles = dir(filePattern);
for i=1:5
    baseFileName = theFiles(i).name;
    fullFileName = fullfile(myFolder, baseFileName);
    RS{i} = audioConvert(30, fullFileName);
    RS{i+5} = audioConvert(45, fullFileName);
    RS{i+10} = audioConvert(50, fullFileName);
    RS{i+15} = audioConvert(60, fullFileName);
    RS{i+20} = audioConvert(70, fullFileName);
    RS{i+25} = audioConvert(80, fullFileName);
    RS{i+30} = audioConvert(100, fullFileName);
    RS{i+35} = audioConvert(110, fullFileName);
    RS{i+40} = audioConvert(150, fullFileName);
    RS{i+45} = audioConvert(160, fullFileName);
end

% Lil Wayne
LW = cell(1,50);
myFolder = '/Users/emilypark/Desktop/WI2018/AMATH482/HW3/test2/Lil Wayne';
filePattern = fullfile(myFolder, '*.mp3');
theFiles = dir(filePattern);
for i=1:5
    baseFileName = theFiles(i).name;
    fullFileName = fullfile(myFolder, baseFileName);
    LW{i} = audioConvert(30, fullFileName);
    LW{i+5} = audioConvert(45, fullFileName);
    LW{i+10} = audioConvert(50, fullFileName);
    LW{i+15} = audioConvert(60, fullFileName);
    LW{i+20} = audioConvert(70, fullFileName);
    LW{i+25} = audioConvert(80, fullFileName);
    LW{i+30} = audioConvert(100, fullFileName);
    LW{i+35} = audioConvert(110, fullFileName);
    LW{i+40} = audioConvert(150, fullFileName);
    LW{i+45} = audioConvert(160, fullFileName);
end

X = zeros(3*50,294921);
for i=1:50
    X(i,:) = DR{i}(:,1);
    X(i+(50),:) = RS{i}(:,1);
    X(i+(100),:) = LW{i}(:,1);
end
x1 = X(1:50,:);
x2 = X(51:100,:);
x3 = X(101:150,:);
[u1,s1,v1] = svd(x1,'econ');
figure(1);
plot(diag(s1),'ro')
[u2,s2,v2] = svd(x2,'econ');
hold on;
plot(diag(s2),'bo')
[u3,s3,v3] = svd(x3,'econ');
plot(diag(s3),'ko')
xlabel('components'); ylabel('singular value'); legend('Drake','Rae Sremmurd','Lil Wayne');

[m,n] = size(X);
[u,s,v] = svd(X,'econ');
%Y = u.' * X;
%Cy=(1/(n-1))*(Y)*(Y.');
%PCs = diag(Cy);
figure(2);plot(diag(s),'ro')
xlabel('components'); ylabel('singular value'); title('Singular Value of Dataset containing severa


dr = v(:,1:50);
rs = v(:,51:100);
lw = v(:,101:150);

o_dr = 0;
o_rs = 0;
o_lw = 0;

for i = 1:100
q1 = randperm(50); q2 = randperm(50); q3 = randperm(50);
train = [dr(q1(1:40),:); rs(q2(1:40),:); lw(q3(1:40),:)];
test = [dr(q1(41:end),:); rs(q2(41:end),:); lw(q3(41:end),:)];
train2 = [ones(40,1);2*ones(40,1);3*ones(40,1)];
ox = [ones(10,1);2*ones(10,1);3*ones(10,1)];

Mdl = fitcnb(train,train2);
prediction = Mdl.predict(test);


for j = 1:30
    if prediction(j)-ox(j) == 0
        if j <= 10
            o_dr = o_dr + 1;
        elseif j <= 20
```

```
            o_rs = o_rs + 1;
        else
            o_lw = o_lw + 1;
        end
    end
end
end

clear all; close all; clc;

L=5; n=220501;
k=(2*pi/L)*[0:n/2-1 -n/2:-1]; ks=fftshift(k);

[y1,fs1] = audioread('jazz.wav');
y1 = (y1(:,1)+y1(:,2))/2;
dt1 = 1/fs1;
cropped1 = y1(30*fs1:35*fs1);
figure(1); subplot(3,1,1); grid on; plot(linspace(30,35,length(cropped1)),cropped1)
title('Jazz');xlabel('Seconds'); ylabel('Amplitude');
%figure(2); plot(psd(spectrum.periodogram,y,'Fs',fs,'NFFT',length(cropped)));
figure(2); subplot(1,3,1); spectrogram(cropped1,[],[],[],fs1,'yaxis'); title('Jazz')

[y2,fs2] = audioread('classical.wav');
y2 = (y2(:,1)+y2(:,2))/2;
dt2 = 1/fs2;
cropped2 = y2(30*fs2:35*fs2);
figure(1); subplot(3,1,2); grid on; plot(linspace(30,35,length(cropped2)),cropped2)
title('Classical');xlabel('Seconds'); ylabel('Amplitude');
%figure(2); plot(psd(spectrum.periodogram,y,'Fs',fs,'NFFT',length(cropped)));
figure(2); subplot(1,3,2); spectrogram(cropped2,[],[],[],fs2,'yaxis'); title('Classical')

[y3,fs3] = audioread('pop.wav');
y3 = (y3(:,1)+y3(:,2))/2;
dt2 = 1/fs3;
cropped3 = y3(30*fs3:35*fs3);
figure(1); subplot(3,1,3); grid on; plot(linspace(30,35,length(cropped3)),cropped3)
title('Pop');xlabel('Seconds'); ylabel('Amplitude');
%figure(2); plot(psd(spectrum.periodogram,y,'Fs',fs,'NFFT',length(cropped)));
figure(2); subplot(1,3,3); spectrogram(cropped3,[],[],[],fs3,'yaxis'); title('Pop')

% figure;
% subplot(3,1,1), plot(linspace(30,35,length(cropped1)),cropped1,'k')
% subplot(3,1,2), plot(linspace(30,35,length(cropped1)),fft(cropped1),'k')
% subplot(3,1,3), plot(ks(1,n/2:n),abs(fftshift(cropped1))/max(abs(cropped1))) %drawnow


clear all; close all; clc;

% Jazz
jazz = cell(1,100);
myFolder = '/Users/emilypark/Desktop/WI2018/AMATH482/HW3/test3/Jazz';
filePattern = fullfile(myFolder, '*.mp3');
theFiles = dir(filePattern);
for i=1:10
    baseFileName = theFiles(i).name;
    fullFileName = fullfile(myFolder, baseFileName);
    jazz{i} = audioConvert(30, fullFileName);
    jazz{i+10} = audioConvert(45, fullFileName);
    jazz{i+20} = audioConvert(50, fullFileName);
    jazz{i+30} = audioConvert(60, fullFileName);
    jazz{i+40} = audioConvert(70, fullFileName);
    jazz{i+50} = audioConvert(80, fullFileName);
    jazz{i+60} = audioConvert(100, fullFileName);
    jazz{i+70} = audioConvert(110, fullFileName);
    jazz{i+80} = audioConvert(150, fullFileName);
    jazz{i+90} = audioConvert(160, fullFileName);
end

% Classical
classical = cell(1,100);
myFolder = '/Users/emilypark/Desktop/WI2018/AMATH482/HW3/test3/Classical';
filePattern = fullfile(myFolder, '*.mp3');
theFiles = dir(filePattern);
for i=1:10
    baseFileName = theFiles(i).name;
    fullFileName = fullfile(myFolder, baseFileName);
    classical{i} = audioConvert(30, fullFileName);
    classical{i+10} = audioConvert(45, fullFileName);
    classical{i+20} = audioConvert(50, fullFileName);
    classical{i+30} = audioConvert(60, fullFileName);
    classical{i+40} = audioConvert(70, fullFileName);
    classical{i+50} = audioConvert(80, fullFileName);
    classical{i+60} = audioConvert(100, fullFileName);
    classical{i+70} = audioConvert(110, fullFileName);
    classical{i+80} = audioConvert(150, fullFileName);
    classical{i+90} = audioConvert(160, fullFileName);
end


% Pop
pop = cell(1,100);
myFolder = '/Users/emilypark/Desktop/WI2018/AMATH482/HW3/test3/Pop';
filePattern = fullfile(myFolder, '*.mp3');
theFiles = dir(filePattern);
for i=1:10
    baseFileName = theFiles(i).name;
    fullFileName = fullfile(myFolder, baseFileName);
    pop{i} = audioConvert(30, fullFileName);
    pop{i+10} = audioConvert(45, fullFileName);
    pop{i+20} = audioConvert(50, fullFileName);
    pop{i+30} = audioConvert(60, fullFileName);
    pop{i+40} = audioConvert(70, fullFileName);
    pop{i+50} = audioConvert(80, fullFileName);
    pop{i+60} = audioConvert(100, fullFileName);
    pop{i+70} = audioConvert(110, fullFileName);
    pop{i+80} = audioConvert(150, fullFileName);
    pop{i+90} = audioConvert(160, fullFileName);
end

X = zeros(3*100,294921);
for i=1:100
    X(i,:) = jazz{i}(:,1);
    X(i+(100),:) = classical{i}(:,1);
    X(i+(200),:) = pop{i}(:,1);
end
x1 = X(1:50,:);
x2 = X(51:100,:);
```

```
x3 = X(101:150,:);
[u1,s1,v1] = svd(x1,'econ');
figure(1);
plot(diag(s1),'ro')
[u2,s2,v2] = svd(x2,'econ');
hold on;
plot(diag(s2),'bo')
[u3,s3,v3] = svd(x3,'econ');
plot(diag(s3),'ko')
xlabel('components'); ylabel('singular value'); legend('Jazz','Classical','Pop');

[m,n] = size(X);
[u,s,v] = svd(X,'econ');
%Y = u.' * X;
%Cy=(1/(n-1))*(Y)*(Y.');
%PCs = diag(Cy);
figure(2);plot(diag(s),'ro')
xlabel('components'); ylabel('singular value'); title('Singular Value of Dataset containing severa


ja = v(:,1:100);
cl = v(:,101:200);
po = v(:,201:300);

o_ja = 0;
o_cl = 0;
o_po = 0;

for i = 1:100
q1 = randperm(100); q2 = randperm(100); q3 = randperm(100);
train = [ja(q1(1:80),:); cl(q2(1:80),:); po(q3(1:80),:)];
test = [ja(q1(81:end),:); cl(q2(81:end),:); po(q3(81:end),:)];
train2 = [ones(80,1);2*ones(80,1);3*ones(80,1)];
ox = [ones(20,1);2*ones(20,1);3*ones(20,1)];

Mdl = fitcnb(train,train2);
prediction = Mdl.predict(test);

for j = 1:60
    if prediction(j)-ox(j) == 0
        if j <= 20
            o_ja = o_ja + 1;
        elseif j <= 40
            o_cl = o_cl + 1;
        else
            o_po = o_po + 1;
        end
    end
end
end
```