AMATH 582 Homework 4

Pavel Zelinsky

Github: <https://github.com/palpalych/MusicAndFaces>

**Abstract**

What kind of things would PCA extract as the principal components of a face dataset? That is to mean – what are the most identifying features of a face? Also, can we identify music bands or genres by using a similar PCA based approach combined with machine learning?

## Introduction and Overview

First the faces – there is a dataset from Yale of a collection of black and white faces. By applying the SVD on all of the faces we can extract what are the principal components. What are the most dominant features of images of human faces? In a similar vein, we have a set of images that are not cropped, how will the dominant features differ between the cropped and uncropped versions?

Onto music. By looking at the SVD of the spectrograms of different pieces of music we can get the primary features across all music. Then we can apply some machine learning algorithms in order to try to solve different problems, for example telling whether a song is Rock vs. Jazz

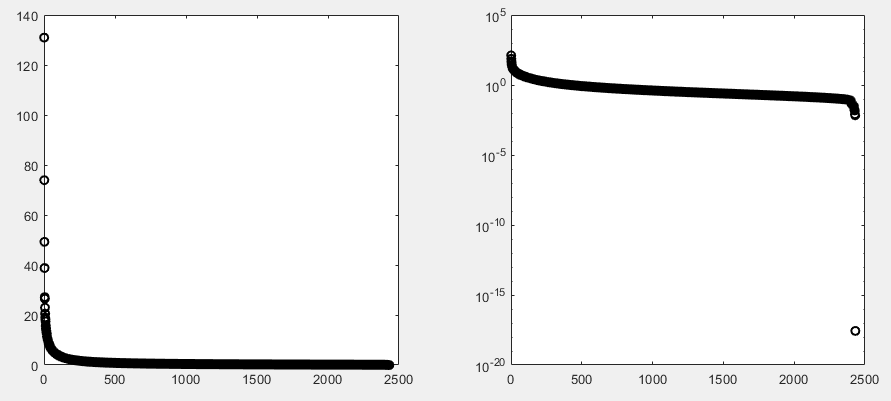
## Theoretical Background

The SVD is a matrix decomposition that breaks any matrix A into two unitary parts (U and V) and a diagonal part ∑

The most important part of the SVD for us here is that if your data is put in as each column of A is a data point, then the first K columns of U\* can be used as a transform for your data into K principal components, with the rows of V cut off at the (K+1)th column are the projections of your data into the principal components:

Given that U and V are unitary, these components are orthogonal to each other. This helps split any data set into the most relevant parts. Note: if your data is instead inserted as rows of A, then everything is still true except U and V are reversed for which is the transform and which is the projections of A. For systems where we can easily make interpretations, for example images of well-known objects, we can look at the different columns of U\* and make some sense of what is the physical meaning of the principal components of the data set

## Algorithm Implementation and Development

For the machine learning algorithms I am using the built in Matlab functions for Naïve Bayes model and Linear Discriminant Analysis. Both of these algorithms are supervised, I chose to use only supervised algorithms since I have labels for my music data. The Naïve Bayes model works by assuming that the data of each classification has a normal distribution, and then the prediction of any new data point is whichever category has the highest probability given the normal distributions from the training set. The Linear Discriminant Analysis algorithm works by putting a line through the data such that the points from different labels projected onto the line are well separated and selects a separation point for whether something is of one category or another

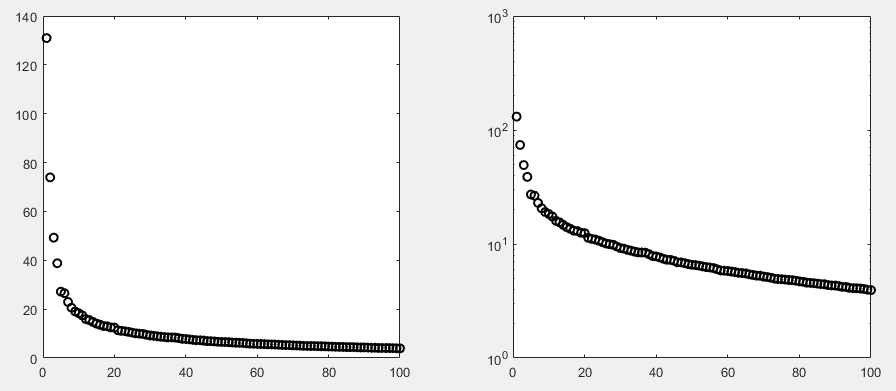
For both implementations one of the most important parts to keep straight is whether the data is columns of A or rows of A, and this can be particularly challenging given that the Matlab sdv function returns V and not V\* and that the Matlab functions for the training algorithms want the data to be in a particular form as well. For the Yale Faces data set I put all the data into columns by taking the grid of pixels and reshaping it into a vector. For the music I did something similar, except in addition I took the spectrogram of each snippet of music, since music tends to be much better defined by the properties of the spectrogram rather than the raw signal itself

Figure 2: First 100 variances of Cropped Yale Faces on a normal and logarithmic scale

Figure 1: Variance of Cropped Yale Faces on a normal and logarithmic scale

## Computational Results

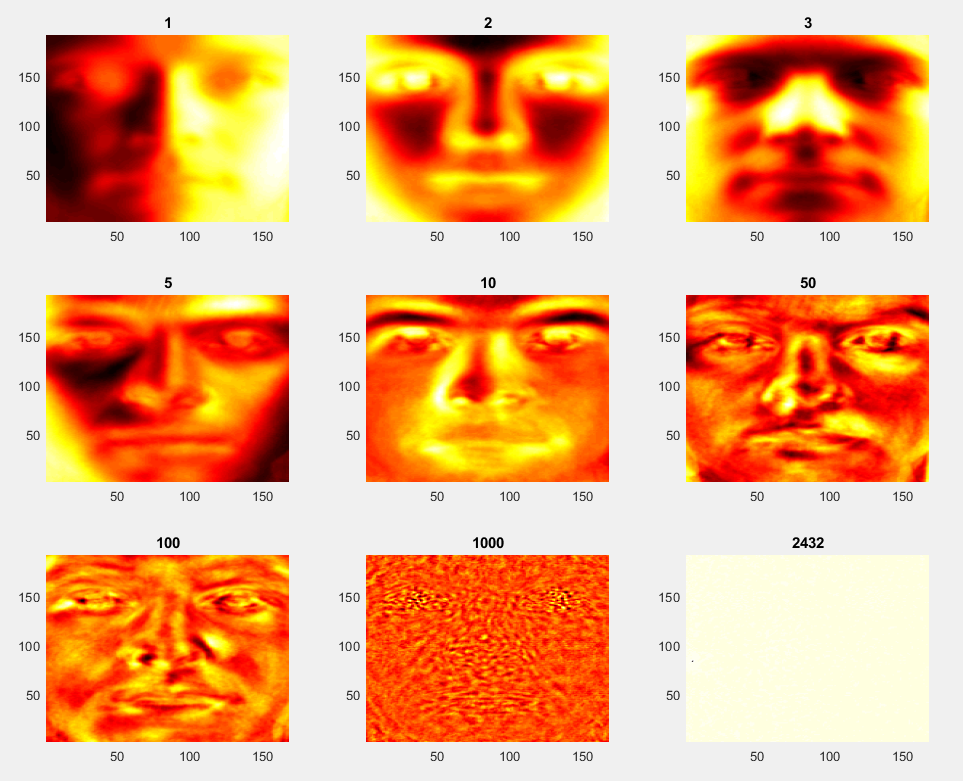
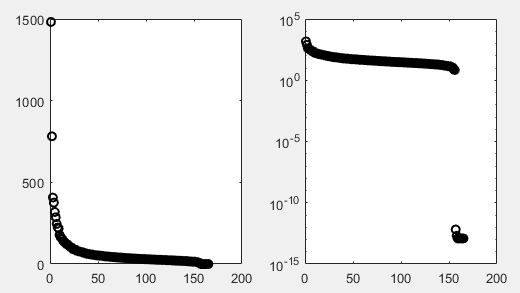
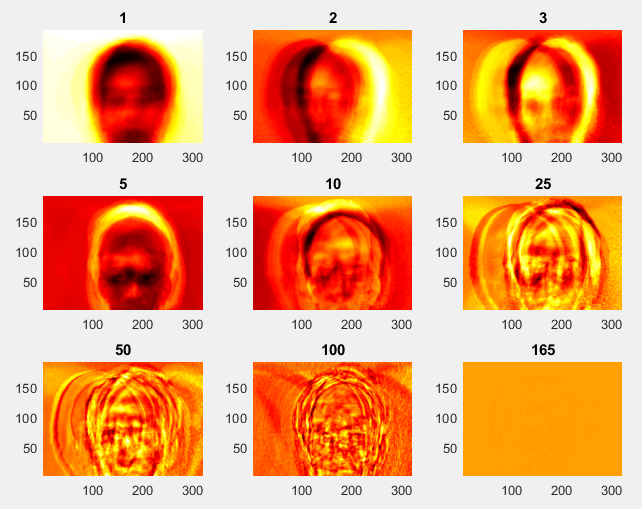
First, a look at the cropped Yale Faces data set. Looking at the variance of the principal components (Figures 1 and 2) we can see that the first 4 components are notably more varied than the rest, but then there is a long tail of component variances. Unfortunately the difference isn’t so large as to confidently ignore all modes past the fourth. What do these modes look like though? Since the modes project images we’re able to take each mode by extracting the particular column of U and by reshaping it into the size of the images view it as one. This gives us an idea of what aspects of the image are considered for the different principal components of our dataset. Looking at Figure 3 we can see a sampling of the different modes. The first mode looks as if it isn’t focused around the details of the face at all but instead around where the lighting is in the picture. This effect shows in Figure 4 where the rank 1 order approximation for the images barely shows a face, and for one of the faces is only the grayness of the overall image. The second mode is starting to pull out some details about the shape of the face, for example if you compare the second mode to the fifth mode one is focused on rounded cheeks versus a slender jaw. Additionally it seems to get a general sense of the nose, mouth and eyes, but without any particular details. In fact even as far as the fifth mode, it seems as if it’s gathering details about the general facial features without targeting any of the details. This shows in Figure 4 as the faces are certainly recognizable as actual faces but they’re blurred and are missing many details, for example all the eyebrows are barely existent. As we get to the higher modes they seem to start to focus more on the particular details of the faces. It’s much harder to tell at this point what particular areas are of interest, but it is interesting to note that in the 1000th mode although it’s hard to tell anything we can see that around the eyes, nose, and mouth there are patterns, indicating that we’re getting some additional information about those details of the face. The last mode seems almost useless as it focuses on what seems to be a single pixel. This is to be expected, since in Figure 1 we saw the last mode plummet in variance compared to the rest

Figure 3: Modes of Cropped Yale Faces dataset

Figure 4: Rank 1, 5, 100, 1000 and original image for random Cropped Yale Faces

Figure 5: Variance of Uncropped Yale Faces on a normal and logarithmic scale

Figure 6: Modes of Uncropped Yale Faces dataset

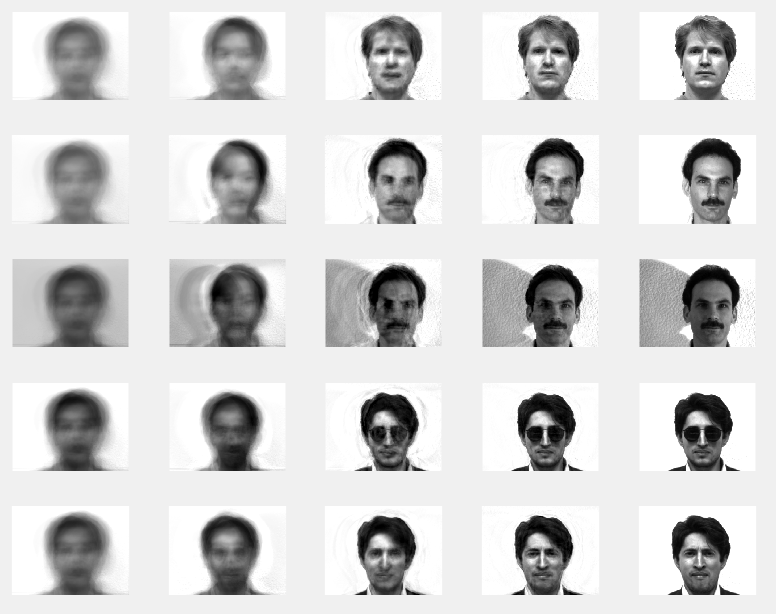
The next interesting point of comparison is the uncropped Yale Faces dataset. Looking at the variance it seems similarly shaped, but there are 2 highly dominant modes. Looking at figure 6, these modes seem to be mostly associated with the location of the head in the image and the background color. Figure 7 shows this nicely since all the left images look like the same head but with different backgrounds, and then the next images over, although rank 5 approximations, mostly have the head repositioned to the correct location without any real details about the faces. Even as we get up to the rank 50 approximation, the image is somewhat blurry. This seems to imply that the PCA is not able to deal well with the uncropped images due to extraneous information, or noise, effectively corrupting the principal components. This can somewhat be seen by looking at the total energy of the system captured by the low rank approximations (Figure 8). The energy plot for the cropped Yale Faces is tighter towards the origin, implying that with a smaller rank approximation it is able to cover more information about the faces.

Figure 7: Rank 1, 5, 50, 100 and original image for random Uncropped Yale Faces

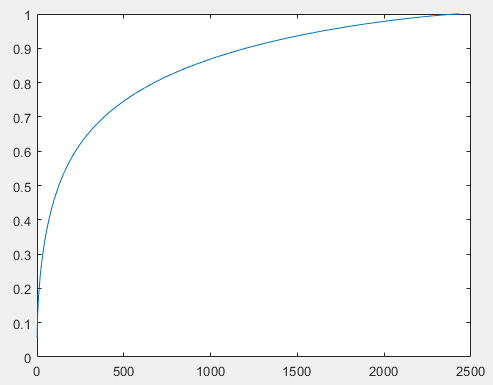
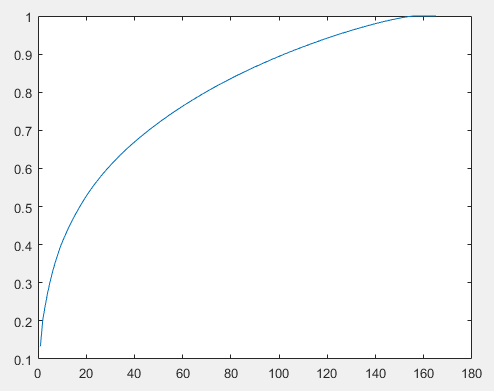


Figure 8: Cropped and Uncropped energy of approximations

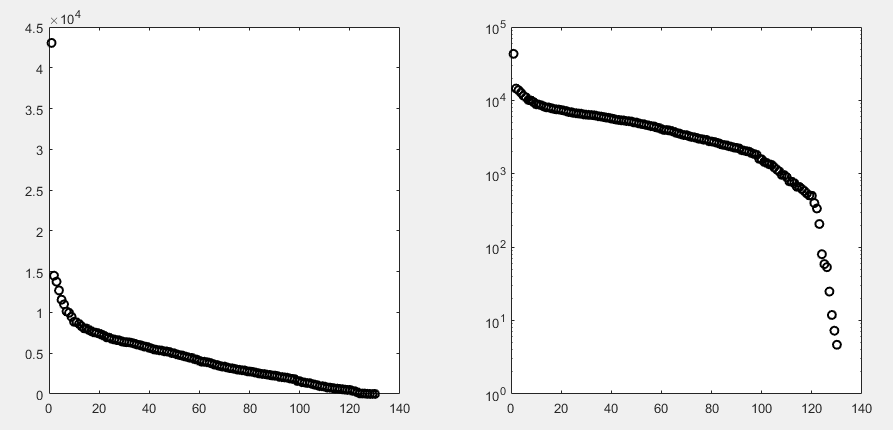
Next, a look into classifying music. I took random 5 second long clips from songs by 3 different artists from 3 different genres, ambient music, blues, and rock, got the spectrogram of each clip, did some PCA and finally split the data into a training set and test set to run through a Naïve Bayes model and an LDA. I additionally took an additional song from each artist that was never part of the training set or test set to see how well the model truly worked for a new song. First, a look at the variances (Figure 9), although the first mode is significantly above the rest, the rest of the modes are clumped and there is no clear good cutoff point to work with. The craziest part – the best training accuracy I was able to get was with 2 modes with consistently around 90% correct, but the best for correctly classifying the truly new songs was using only the 1st mode at around 70% accuracy. The training algorithms performed relatively similarly with LDA consistently slightly more accurate. One big note on the training algorithms – as I would increase the number of modes considered, Naïve Bayes would perform notable worse than LDA on the never seen song. In terms of why this is still quite inaccurate – ambient music wasn’t the best choice of a genre to select for having strongly distinct music genres. It is a genre that is meant to be background music, but uses a wide variety of instruments and music styles, and especially the artist I had randomly selected seems to have many aspects of blues within the songs. This caused frequent mistakes for the algorithms in ambient vs blues. On the other hand, the algorithms would almost never confuse rock and blues

Figure 9: Variance of rock, blues, and ambient music

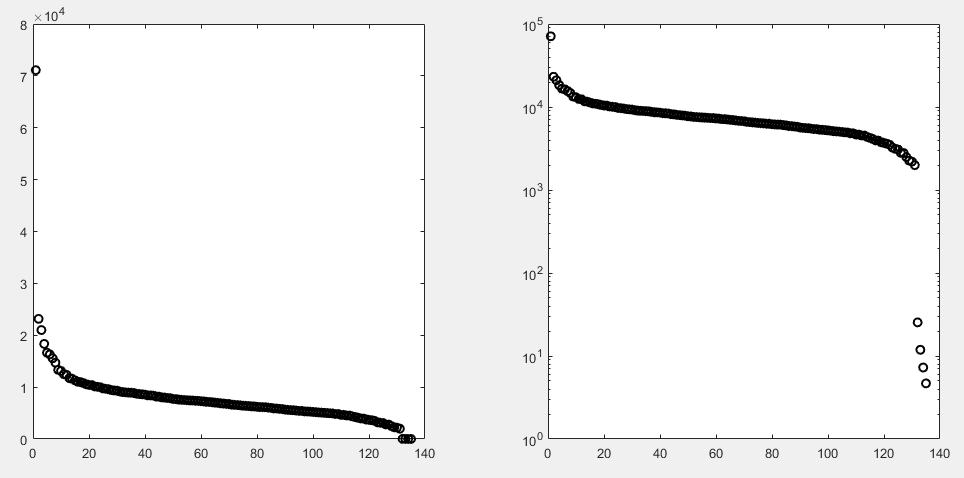
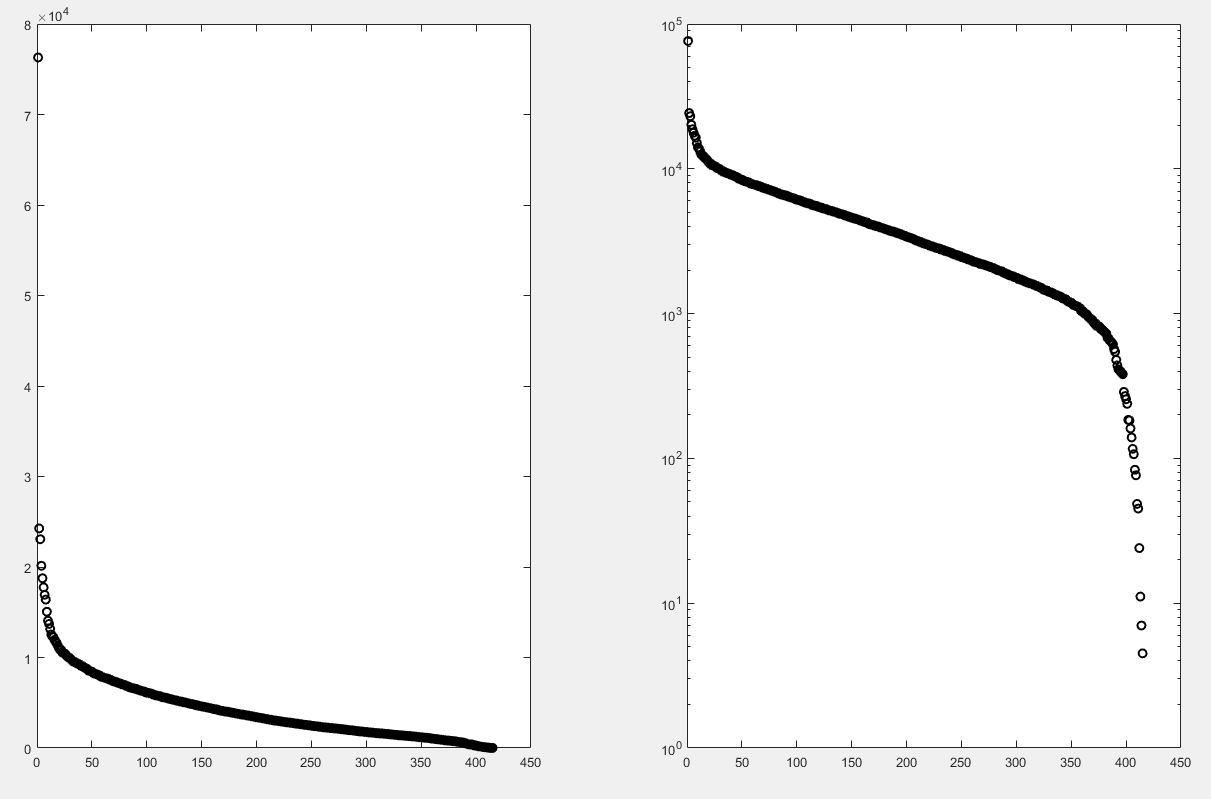
Not having learned enough from my mistakes, I then took 5 second long clips from songs by 3 different random rock artists. This is expected to be tough for a learning algorithm to distinguish since rock songs will generally seem alike and the more minute details will point out the difference between bands. When looking at the variance (Figure 10) several interesting things pop out – there are 5 groupings of variances. The first grouping is just the first mode on its own, then another with the second and third mode, another one of modes 4 through 8, a big group of nearly the rest of the modes, and then the last 4 modes. This is quite an interesting pattern as it gives some potential cutoff points to consider when selecting which principal components to use for training. The surprising part is even with only 3 modes I was seeing a consistent 80% or higher accuracy on the training set, and 86% accuracy on the never before seen songs. With 8 modes I was always seeing 100% accuracy on never before seen songs using the LDA algorithm. Naïve Bayes was able to consistently get 90% accuracy and both had around 86% accuracy on the test set, but LDA was able to consistently correctly nail down the bands correctly. This result is somewhat unexpected – classifying bands within a genre should be more difficult than classifying bands across genres since different genres tend to have vastly differing styles and patterns. In this case it made sense once I listened to the rock music, each of the bands had distinctly different styles (folk style vs. hard rock) or distinctly different instruments used (guitar vs. electric guitar). Ultimately this meant that to the machine learning algorithms these were more distinctive from each other than the ambient music from blues and rock

Figure 11: Variance of jazz, rock, and ambient music

Figure 10: Variance of different rock music

Thinking I learned from my mistakes, I next took 5 seconds clips from songs by 3 different artists for each genre, but I changed genres to jazz, rock, and ambient. My goal this time was to correctly categorize a new band for which genre it belongs to. Looking at the variance (Figure 11) we once again have one mode well above the rest, followed by two clumped together, and then another 5 clumped together. Using only the first mode gave accuracy of only around 65% for test set and 50% for the never seen songs. The best I managed to obtain was with 3 modes getting consistently around 75% accuracy for the test set and around 60% accuracy for the never seen songs using the Naïve Bayes model, but substantially worse accuracy with the LDA. Note – going up drastically in the number of modes would sometimes help with the test set accuracy, but at the end of the day it would not help the accuracy of categorizing the never seen data set. This task being more challenging to do accurately than categorizing specific bands across genres is expected, since a new band within a genre will still have stylistic differences that can be akin to some other genre for the purposes of the algorithm

## Summary and Conclusion

The biggest take-away here – when picking genres of music for classification stay away from ambient music unless you want a challenge. It’s quite a generic category that uses instruments and styles from many other genres, so much so that categorizing different styles of rock music turned out to be simpler than categorizing rock vs. ambient

Additionally, the SVD is able to provide quick insight into data, but in order to be most useful some preprocessing to make the data more focused around the problem at hand is needed

Lastly – math is cool

Appendix A – MATLAB Functions and Code Notes:

Semilogy – plot on a logarithmic scale

Pcolor, shading, set, xlabel, ylabel, title, colormap, figure, subplot – functions used for plotting, organizing, and labeling the data

Image, implay – functions for looking at images and playing videos. This was highly useful as Matlab allows looking at videos frame by frame, and even get the exact RGB values of specific frames

Svd – best function in Matlab. Computes the SVD. The ‘econ’ flag is especially useful

Diag – extract just the diagonal elements of a matrix, useful when handling the SVD

Randperm – generate a random permutation vector. This is helpful for randomly rearranging data for a training and test set

Fitcnd, predict – train a Naïve Bayes model and make a prediction from the model

Classify – classify some given new data points via the given training set. Note that this doesn’t train a model but is called with every new data point

Notes on code:

I split the homework into Homework4Faces and Homework4Music as they were quite different in what they did and how. The code is split into sections such that once a certain music data set is loaded, it is easy to try different start and end modes

Appendix B – MATLAB Code:

Homework4Faces.m

%% Start with a clean slate

clear all; close all; clc

%% cropped images

directories = dir('CroppedYale');

allFaces = [];

for i=1:length(directories)

if directories(i).name == "." | directories(i).name == ".."

continue

end

images = dir(strcat(directories(i).folder, '\', directories(i).name));

for j=1:length(images)

if images(j).name == "." | images(j).name == ".."

continue

end

face = imread(strcat(images(j).folder, '\', images(j).name));

allFaces = [allFaces reshape(face,192\*168,1)];

end

end

% normalize

original\_Faces = allFaces;

data\_size = size(allFaces);

allFaces = double(allFaces)./(data\_size(2)-1);

mean\_Faces = mean(allFaces);

allFaces = allFaces - mean\_Faces;

[u,s,v] = svd(allFaces,'econ');

sig=diag(s);

figure(1)

subplot(1,2,1), plot(sig,'ko','Linewidth',[1.5])

subplot(1,2,2), semilogy(sig,'ko','Linewidth',[1.5])

figure(2)

subplot(1,2,1), plot(sig,'ko','Linewidth',[1.5])

set(gca,'Xlim',[0 100])

subplot(1,2,2), semilogy(sig,'ko','Linewidth',[1.5])

set(gca,'Xlim',[0 100])

energy = zeros(1,data\_size(2));

for i=1:data\_size(2)

energy(i)=sum(sig(1:i))/sum(sig);

end

figure(3)

plot(energy)

figure(4)

num\_modes = [1 5 100 1000];

for i=1:5

for j=1:length(num\_modes)

subplot(5,5,i\*5+j-5)

approx1 = (u(:,1:num\_modes(j))\*s(1:num\_modes(j),1:num\_modes(j))\*v(:,1:num\_modes(j))' + mean\_Faces).\*(data\_size(2)-1);

approx1 = uint8(approx1);

imshow(reshape(approx1(:,i\*23),192,168))

end

subplot(5,5,i\*5)

imshow(reshape(original\_Faces(:,i\*23),192,168))

end

figure(5)

num\_modes = [1 2 3 5 10 50 100 1000 2432];

for j=1:length(num\_modes)

subplot(3,3,j)

% taken from code provided in class - otherwise images come out flipped

ut1=reshape(u(:,num\_modes(j)),192,168);

ut2=ut1(192:-1:1,:);

h = pcolor(ut2);

colormap(hot)

% remove edges, otherwise image details are hidden

set(h, 'EdgeColor', 'none');

title(num\_modes(j))

end

%% uncropped images

images = dir('yalefaces');

allUFaces = [];

for j=1:length(images)

if images(j).name == "." | images(j).name == ".."

continue

end

face = imread(strcat(images(j).folder, '\', images(j).name));

allUFaces = [allUFaces reshape(face,243\*320,1)];

end

% normalize

original\_UFaces = allUFaces;

Udata\_size = size(allUFaces);

allUFaces = double(allUFaces)./(Udata\_size(2)-1);

mean\_UFaces = mean(allUFaces);

allUFaces = allUFaces - mean\_UFaces;

[uu,us,uv] = svd(allUFaces,'econ');

usig=diag(us);

figure(6)

subplot(1,2,1), plot(usig,'ko','Linewidth',[1.5])

subplot(1,2,2), semilogy(usig,'ko','Linewidth',[1.5])

figure(7)

subplot(1,2,1), plot(usig,'ko','Linewidth',[1.5])

set(gca,'Xlim',[0 100])

subplot(1,2,2), semilogy(usig,'ko','Linewidth',[1.5])

set(gca,'Xlim',[0 100])

uenergy = zeros(1,Udata\_size(2));

for i=1:Udata\_size(2)

uenergy(i)=sum(usig(1:i))/sum(usig);

end

figure(8)

plot(uenergy)

figure(9)

num\_modes = [1 5 50 100];

for i=1:5

for j=1:length(num\_modes)

subplot(5,5,i\*5+j-5)

approx1 = (uu(:,1:num\_modes(j))\*us(1:num\_modes(j),1:num\_modes(j))\*uv(:,1:num\_modes(j))' + mean\_UFaces).\*(Udata\_size(2)-1);

approx1 = uint8(approx1);

imshow(reshape(approx1(:,i\*6),243,320))

end

subplot(5,5,i\*5)

imshow(reshape(original\_UFaces(:,i\*6),243,320))

end

figure(10)

num\_modes = [1 2 3 5 10 25 50 100 165];

for j=1:length(num\_modes)

subplot(3,3,j)

% taken from code provided in class - otherwise images come out flipped

ut1=reshape(uu(:,num\_modes(j)),243,320);

ut2=ut1(192:-1:1,:);

h = pcolor(ut2);

colormap(hot)

% remove edges, otherwise image details are hidden

set(h, 'EdgeColor', 'none');

title(num\_modes(j))

end

Homework4Music.m

%% Start with a clean slate

clear all; close all; clc

%% Band Classification

songsFolder = dir('Music\Ambient');

ambientSongs = [];

for i=1:length(songsFolder)

if songsFolder(i).name == "." | songsFolder(i).name == ".."

continue

end

% I'm assuming that the left vs right stereo aren't noticeably important for

% these clips. Additionally reducing the signal frequency for a more

% manageable amount of data

[m,Fs] = audioread(strcat(songsFolder(i).folder, '\', songsFolder(i).name));

ambientSongs = [ambientSongs reshape(spectrogram(m(1:2:Fs\*5+1,1)),16385\*8,1) ...

reshape(spectrogram(m(Fs\*15:2:Fs\*20+1,1)),16385\*8,1) ...

reshape(spectrogram(m(Fs\*30:2:Fs\*35+1,1)),16385\*8,1) ...

reshape(spectrogram(m(Fs\*55:2:Fs\*60+1,1)),16385\*8,1) ...

reshape(spectrogram(m(Fs\*65:2:Fs\*70+1,1)),16385\*8,1)];

end

songsFolder = dir('Music\Blues');

bluesSongs = [];

for i=1:length(songsFolder)

if songsFolder(i).name == "." | songsFolder(i).name == ".."

continue

end

[m,Fs] = audioread(strcat(songsFolder(i).folder, '\', songsFolder(i).name));

bluesSongs = [bluesSongs reshape(spectrogram(m(1:2:Fs\*5+1,1)),16385\*8,1) ...

reshape(spectrogram(m(Fs\*15:2:Fs\*20+1,1)),16385\*8,1) ...

reshape(spectrogram(m(Fs\*30:2:Fs\*35+1,1)),16385\*8,1) ...

reshape(spectrogram(m(Fs\*55:2:Fs\*60+1,1)),16385\*8,1) ...

reshape(spectrogram(m(Fs\*65:2:Fs\*70+1,1)),16385\*8,1)];

end

songsFolder = dir('Music\Rock');

rockSongs = [];

for i=1:length(songsFolder)

if songsFolder(i).name == "." | songsFolder(i).name == ".."

continue

end

[m,Fs] = audioread(strcat(songsFolder(i).folder, '\', songsFolder(i).name));

rockSongs = [rockSongs reshape(spectrogram(m(1:2:Fs\*5+1,1)),16385\*8,1) ...

reshape(spectrogram(m(Fs\*15:2:Fs\*20+1,1)),16385\*8,1) ...

reshape(spectrogram(m(Fs\*30:2:Fs\*35+1,1)),16385\*8,1) ...

reshape(spectrogram(m(Fs\*55:2:Fs\*60+1,1)),16385\*8,1) ...

reshape(spectrogram(m(Fs\*65:2:Fs\*70+1,1)),16385\*8,1)];

end

allSongs = abs([ambientSongs bluesSongs rockSongs]);

mean\_allSongs = mean(allSongs);

allSongsCentered = allSongs - mean\_allSongs;

[u,s,v] = svd(allSongsCentered,'econ');

sig=diag(s);

figure(1)

subplot(1,2,1), plot(sig,'ko','Linewidth',[1.5])

subplot(1,2,2), semilogy(sig,'ko','Linewidth',[1.5])

%%

startmode = 1;

endmodes = 80;

Transform = s(startmode:endmodes,startmode:endmodes)\u(:,startmode:endmodes)';

random\_ambient = randperm(40);

random\_blues = randperm(45);

random\_rock = randperm(45);

ambient = v(1:40,startmode:endmodes);

blues = v(41:85,startmode:endmodes);

rock = v(86:end,startmode:endmodes);

xtrain=[ambient(random\_ambient(1:35),:); blues(random\_blues(1:40),:); rock(random\_rock(1:40),:)];

xtest=[ambient(random\_ambient(36:end),:); blues(random\_blues(41:end),:); rock(random\_rock(41:end),:)];

xexpected=string(zeros(15,1));

xexpected(1:5)="ambient";

xexpected(6:10)="blues";

xexpected(11:15)="rock";

ctrain=string(zeros(115,1));

ctrain(1:35)="ambient";

ctrain(36:75)="blues";

ctrain(76:end)="rock";

% try both a Gaussian Naive Bayes model and an LDA

nb=fitcnb(xtrain,ctrain);

trainAccuracyBayes = sum(nb.predict(xtest)==xexpected)/15

trainAccuracyLDA = sum(classify(xtest,xtrain,ctrain)==xexpected)/15

correctBayes = 0;

correctLda = 0;

% take a completely unused Ambient song and test against it

[m,Fs] = audioread('.\Music\Test\Ambient.mp3');

testAmbient = [reshape(spectrogram(m(1:2:Fs\*5+1,1)),16385\*8,1) ...

reshape(spectrogram(m(Fs\*15:2:Fs\*20+1,1)),16385\*8,1) ...

reshape(spectrogram(m(Fs\*30:2:Fs\*35+1,1)),16385\*8,1) ...

reshape(spectrogram(m(Fs\*55:2:Fs\*60+1,1)),16385\*8,1) ...

reshape(spectrogram(m(Fs\*65:2:Fs\*70+1,1)),16385\*8,1)];

testAmbient = abs(testAmbient);

testAmbient = testAmbient - mean(testAmbient);

transformedAmbient = (Transform \* testAmbient)';

correctBayes = correctBayes + sum(nb.predict(transformedAmbient)=="ambient");

correctLda = correctLda + sum(classify(transformedAmbient,xtrain,ctrain)=="ambient");

% Repeat for a Rock song

[m,Fs] = audioread('.\Music\Test\Rock.mp3');

testRock = [reshape(spectrogram(m(1:2:Fs\*5+1,1)),16385\*8,1) ...

reshape(spectrogram(m(Fs\*15:2:Fs\*20+1,1)),16385\*8,1) ...

reshape(spectrogram(m(Fs\*30:2:Fs\*35+1,1)),16385\*8,1) ...

reshape(spectrogram(m(Fs\*55:2:Fs\*60+1,1)),16385\*8,1) ...

reshape(spectrogram(m(Fs\*65:2:Fs\*70+1,1)),16385\*8,1)];

testRock = abs(testRock);

testRock = testRock - mean(testRock);

transformedRock = (Transform \* testRock)';

correctBayes = correctBayes + sum(nb.predict(transformedRock)=="rock");

correctLda = correctLda + sum(classify(transformedRock,xtrain,ctrain)=="rock");

% Repeat for a Blues song

[m,Fs] = audioread('.\Music\Test\Blues.mp3');

testBlues = [reshape(spectrogram(m(1:2:Fs\*5+1,1)),16385\*8,1) ...

reshape(spectrogram(m(Fs\*15:2:Fs\*20+1,1)),16385\*8,1) ...

reshape(spectrogram(m(Fs\*30:2:Fs\*35+1,1)),16385\*8,1) ...

reshape(spectrogram(m(Fs\*55:2:Fs\*60+1,1)),16385\*8,1) ...

reshape(spectrogram(m(Fs\*65:2:Fs\*70+1,1)),16385\*8,1)];

testBlues = abs(testBlues);

testBlues = testBlues - mean(testBlues);

transformedBlues = (Transform \* testBlues)';

correctBayes = correctBayes + sum(nb.predict(transformedBlues)=="blues");

correctLda = correctLda + sum(classify(transformedBlues,xtrain,ctrain)=="blues");

BayesAccuracy = correctBayes / 15

LDAAccuracy = correctLda / 15

%% Classification within genre

songsFolder = dir('Music\Rock2');

rock2Songs = [];

for i=1:length(songsFolder)

if songsFolder(i).name == "." | songsFolder(i).name == ".."

continue

end

[m,Fs] = audioread(strcat(songsFolder(i).folder, '\', songsFolder(i).name));

rock2Songs = [rock2Songs reshape(spectrogram(m(1:2:Fs\*5+1,1)),16385\*8,1) ...

reshape(spectrogram(m(Fs\*15:2:Fs\*20+1,1)),16385\*8,1) ...

reshape(spectrogram(m(Fs\*30:2:Fs\*35+1,1)),16385\*8,1) ...

reshape(spectrogram(m(Fs\*55:2:Fs\*60+1,1)),16385\*8,1) ...

reshape(spectrogram(m(Fs\*65:2:Fs\*70+1,1)),16385\*8,1)];

end

songsFolder = dir('Music\Rock3');

rock3Songs = [];

for i=1:length(songsFolder)

if songsFolder(i).name == "." | songsFolder(i).name == ".."

continue

end

[m,Fs] = audioread(strcat(songsFolder(i).folder, '\', songsFolder(i).name));

rock3Songs = [rock3Songs reshape(spectrogram(m(1:2:Fs\*5+1,1)),16385\*8,1) ...

reshape(spectrogram(m(Fs\*15:2:Fs\*20+1,1)),16385\*8,1) ...

reshape(spectrogram(m(Fs\*30:2:Fs\*35+1,1)),16385\*8,1) ...

reshape(spectrogram(m(Fs\*55:2:Fs\*60+1,1)),16385\*8,1) ...

reshape(spectrogram(m(Fs\*65:2:Fs\*70+1,1)),16385\*8,1)];

end

allSongs = abs([rock3Songs rock2Songs rockSongs]);

mean\_allSongs = mean(allSongs);

allSongsCentered = allSongs - mean\_allSongs;

[ru,rs,rv] = svd(allSongsCentered,'econ');

sig=diag(rs);

figure(2)

subplot(1,2,1), plot(sig,'ko','Linewidth',[1.5])

subplot(1,2,2), semilogy(sig,'ko','Linewidth',[1.5])

%%

startmode = 1;

endmodes = 8;

Transform = rs(startmode:endmodes,startmode:endmodes)\ru(:,startmode:endmodes)';

random\_rock3 = randperm(45);

random\_rock2 = randperm(45);

random\_rock = randperm(45);

rock3 = rv(1:45,startmode:endmodes);

rock2 = rv(46:90,startmode:endmodes);

rock = rv(91:end,startmode:endmodes);

xtrain=[rock3(random\_rock3(1:40),:); rock2(random\_rock2(1:40),:); rock(random\_rock(1:40),:)];

xtest=[rock3(random\_rock3(41:end),:); rock2(random\_rock2(41:end),:); rock(random\_rock(41:end),:)];

xexpected=string(zeros(15,1));

xexpected(1:5)="rock3";

xexpected(6:10)="rock2";

xexpected(11:15)="rock";

ctrain=string(zeros(120,1));

ctrain(1:40)="rock3";

ctrain(41:80)="rock2";

ctrain(81:end)="rock";

% try both a Gaussian Naive Bayes model and an LDA

nb=fitcnb(xtrain,ctrain);

trainAccuracyBayes = sum(nb.predict(xtest)==xexpected)/15

trainAccuracyLDA = sum(classify(xtest,xtrain,ctrain)==xexpected)/15

correctBayes = 0;

correctLda = 0;

% take a completely unused Rock song and test against it

[m,Fs] = audioread('.\Music\Test\Rock.mp3');

testRock = [reshape(spectrogram(m(1:2:Fs\*5+1,1)),16385\*8,1) ...

reshape(spectrogram(m(Fs\*15:2:Fs\*20+1,1)),16385\*8,1) ...

reshape(spectrogram(m(Fs\*30:2:Fs\*35+1,1)),16385\*8,1) ...

reshape(spectrogram(m(Fs\*55:2:Fs\*60+1,1)),16385\*8,1) ...

reshape(spectrogram(m(Fs\*65:2:Fs\*70+1,1)),16385\*8,1)];

testRock = abs(testRock);

testRock = testRock - mean(testRock);

transformedRock = (Transform \* testRock)';

correctBayes = correctBayes + sum(nb.predict(transformedRock)=="rock");

correctLda = correctLda + sum(classify(transformedRock,xtrain,ctrain)=="rock");

% Repeat for an unused Rock2 song

[m,Fs] = audioread('.\Music\Test\Rock2.mp3');

testRock = [reshape(spectrogram(m(1:2:Fs\*5+1,1)),16385\*8,1) ...

reshape(spectrogram(m(Fs\*15:2:Fs\*20+1,1)),16385\*8,1) ...

reshape(spectrogram(m(Fs\*30:2:Fs\*35+1,1)),16385\*8,1) ...

reshape(spectrogram(m(Fs\*55:2:Fs\*60+1,1)),16385\*8,1) ...

reshape(spectrogram(m(Fs\*65:2:Fs\*70+1,1)),16385\*8,1)];

testRock = abs(testRock);

testRock = testRock - mean(testRock);

transformedRock = (Transform \* testRock)';

correctBayes = correctBayes + sum(nb.predict(transformedRock)=="rock2");

correctLda = correctLda + sum(classify(transformedRock,xtrain,ctrain)=="rock2");

% Repeat for an unused Rock3 song

[m,Fs] = audioread('.\Music\Test\Rock3.mp3');

testRock = [reshape(spectrogram(m(1:2:Fs\*5+1,1)),16385\*8,1) ...

reshape(spectrogram(m(Fs\*15:2:Fs\*20+1,1)),16385\*8,1) ...

reshape(spectrogram(m(Fs\*30:2:Fs\*35+1,1)),16385\*8,1) ...

reshape(spectrogram(m(Fs\*55:2:Fs\*60+1,1)),16385\*8,1) ...

reshape(spectrogram(m(Fs\*65:2:Fs\*70+1,1)),16385\*8,1)];

testRock = abs(testRock);

testRock = testRock - mean(testRock);

transformedRock = (Transform \* testRock)';

correctBayes = correctBayes + sum(nb.predict(transformedRock)=="rock3");

correctLda = correctLda + sum(classify(transformedRock,xtrain,ctrain)=="rock3");

BayesAccuracy = correctBayes / 15

LDAAccuracy = correctLda / 15

%% Genre Classification

songsFolder = dir('Music\Jazz');

jazzSongs = [];

for i=1:length(songsFolder)

if songsFolder(i).name == "." | songsFolder(i).name == ".."

continue

end

[m,Fs] = audioread(strcat(songsFolder(i).folder, '\', songsFolder(i).name));

jazzSongs = [jazzSongs reshape(spectrogram(m(1:2:Fs\*5+1,1)),16385\*8,1) ...

reshape(spectrogram(m(Fs\*15:2:Fs\*20+1,1)),16385\*8,1) ...

reshape(spectrogram(m(Fs\*30:2:Fs\*35+1,1)),16385\*8,1) ...

reshape(spectrogram(m(Fs\*55:2:Fs\*60+1,1)),16385\*8,1) ...

reshape(spectrogram(m(Fs\*65:2:Fs\*70+1,1)),16385\*8,1)];

end

songsFolder = dir('Music\Jazz2');

jazz2Songs = [];

for i=1:length(songsFolder)

if songsFolder(i).name == "." | songsFolder(i).name == ".."

continue

end

[m,Fs] = audioread(strcat(songsFolder(i).folder, '\', songsFolder(i).name));

jazz2Songs = [jazz2Songs reshape(spectrogram(m(1:2:Fs\*5+1,1)),16385\*8,1) ...

reshape(spectrogram(m(Fs\*15:2:Fs\*20+1,1)),16385\*8,1) ...

reshape(spectrogram(m(Fs\*30:2:Fs\*35+1,1)),16385\*8,1) ...

reshape(spectrogram(m(Fs\*55:2:Fs\*60+1,1)),16385\*8,1) ...

reshape(spectrogram(m(Fs\*65:2:Fs\*70+1,1)),16385\*8,1)];

end

songsFolder = dir('Music\Jazz3');

jazz3Songs = [];

for i=1:length(songsFolder)

if songsFolder(i).name == "." | songsFolder(i).name == ".."

continue

end

[m,Fs] = audioread(strcat(songsFolder(i).folder, '\', songsFolder(i).name));

jazz3Songs = [jazz3Songs reshape(spectrogram(m(1:2:Fs\*5+1,1)),16385\*8,1) ...

reshape(spectrogram(m(Fs\*15:2:Fs\*20+1,1)),16385\*8,1) ...

reshape(spectrogram(m(Fs\*30:2:Fs\*35+1,1)),16385\*8,1) ...

reshape(spectrogram(m(Fs\*55:2:Fs\*60+1,1)),16385\*8,1)];

end

songsFolder = dir('Music\Ambient2');

ambient2Songs = [];

for i=1:length(songsFolder)

if songsFolder(i).name == "." | songsFolder(i).name == ".."

continue

end

[m,Fs] = audioread(strcat(songsFolder(i).folder, '\', songsFolder(i).name));

ambient2Songs = [ambient2Songs reshape(spectrogram(m(1:2:Fs\*5+1,1)),16385\*8,1) ...

reshape(spectrogram(m(Fs\*15:2:Fs\*20+1,1)),16385\*8,1) ...

reshape(spectrogram(m(Fs\*30:2:Fs\*35+1,1)),16385\*8,1) ...

reshape(spectrogram(m(Fs\*55:2:Fs\*60+1,1)),16385\*8,1) ...

reshape(spectrogram(m(Fs\*65:2:Fs\*70+1,1)),16385\*8,1)];

end

songsFolder = dir('Music\Ambient3');

ambient3Songs = [];

for i=1:length(songsFolder)

if songsFolder(i).name == "." | songsFolder(i).name == ".."

continue

end

[m,Fs] = audioread(strcat(songsFolder(i).folder, '\', songsFolder(i).name));

ambient3Songs = [ambient3Songs reshape(spectrogram(m(1:2:Fs\*5+1,1)),16385\*8,1) ...

reshape(spectrogram(m(Fs\*15:2:Fs\*20+1,1)),16385\*8,1) ...

reshape(spectrogram(m(Fs\*30:2:Fs\*35+1,1)),16385\*8,1) ...

reshape(spectrogram(m(Fs\*55:2:Fs\*60+1,1)),16385\*8,1) ...

reshape(spectrogram(m(Fs\*65:2:Fs\*70+1,1)),16385\*8,1)];

end

allSongs = abs([jazzSongs jazz2Songs jazz3Songs rock3Songs rock2Songs rockSongs ambientSongs ambient2Songs ambient3Songs]);

mean\_allSongs = mean(allSongs);

allSongsCentered = allSongs - mean\_allSongs;

[ru,rs,rv] = svd(allSongsCentered,'econ');

sig=diag(rs);

figure(3)

subplot(1,2,1), plot(sig,'ko','Linewidth',[1.5])

subplot(1,2,2), semilogy(sig,'ko','Linewidth',[1.5])

%%

startmode = 1;

endmodes = 250;

Transform = rs(startmode:endmodes,startmode:endmodes)\ru(:,startmode:endmodes)';

random\_jazz = randperm(140);

random\_rock = randperm(135);

random\_ambient = randperm(140);

jazz = rv(1:140,startmode:endmodes);

rock = rv(141:275,startmode:endmodes);

ambient = rv(276:end,startmode:endmodes);

xtrain=[jazz(random\_jazz(1:120),:); rock(random\_rock(1:120),:); ambient(random\_ambient(1:120),:)];

xtest=[jazz(random\_jazz(121:end),:); rock(random\_rock(121:end),:); ambient(random\_ambient(121:end),:)];

xexpected=string(zeros(55,1));

xexpected(1:20)="jazz";

xexpected(21:35)="rock";

xexpected(36:55)="ambient";

ctrain=string(zeros(360,1));

ctrain(1:120)="jazz";

ctrain(121:240)="rock";

ctrain(241:end)="ambient";

% try both a Gaussian Naive Bayes model and an LDA

nb=fitcnb(xtrain,ctrain);

trainAccuracyBayes = sum(nb.predict(xtest)==xexpected)/55

trainAccuracyLDA = sum(classify(xtest,xtrain,ctrain)==xexpected)/55

correctBayes = 0;

correctLda = 0;

% take a completely unused Rock song and test against it

[m,Fs] = audioread('.\Music\Test\Rock.mp3');

testRock = [reshape(spectrogram(m(1:2:Fs\*5+1,1)),16385\*8,1) ...

reshape(spectrogram(m(Fs\*15:2:Fs\*20+1,1)),16385\*8,1) ...

reshape(spectrogram(m(Fs\*30:2:Fs\*35+1,1)),16385\*8,1) ...

reshape(spectrogram(m(Fs\*55:2:Fs\*60+1,1)),16385\*8,1) ...

reshape(spectrogram(m(Fs\*65:2:Fs\*70+1,1)),16385\*8,1)];

testRock = abs(testRock);

testRock = testRock - mean(testRock);

transformedRock = (Transform \* testRock)';

correctBayes = correctBayes + sum(nb.predict(transformedRock)=="rock");

correctLda = correctLda + sum(classify(transformedRock,xtrain,ctrain)=="rock");

% Repeat for an unused Rock2 song

[m,Fs] = audioread('.\Music\Test\Rock2.mp3');

testRock = [reshape(spectrogram(m(1:2:Fs\*5+1,1)),16385\*8,1) ...

reshape(spectrogram(m(Fs\*15:2:Fs\*20+1,1)),16385\*8,1) ...

reshape(spectrogram(m(Fs\*30:2:Fs\*35+1,1)),16385\*8,1) ...

reshape(spectrogram(m(Fs\*55:2:Fs\*60+1,1)),16385\*8,1) ...

reshape(spectrogram(m(Fs\*65:2:Fs\*70+1,1)),16385\*8,1)];

testRock = abs(testRock);

testRock = testRock - mean(testRock);

transformedRock = (Transform \* testRock)';

correctBayes = correctBayes + sum(nb.predict(transformedRock)=="rock2");

correctLda = correctLda + sum(classify(transformedRock,xtrain,ctrain)=="rock2");

% Repeat for an unused Rock3 song

[m,Fs] = audioread('.\Music\Test\Rock3.mp3');

testRock = [reshape(spectrogram(m(1:2:Fs\*5+1,1)),16385\*8,1) ...

reshape(spectrogram(m(Fs\*15:2:Fs\*20+1,1)),16385\*8,1) ...

reshape(spectrogram(m(Fs\*30:2:Fs\*35+1,1)),16385\*8,1) ...

reshape(spectrogram(m(Fs\*55:2:Fs\*60+1,1)),16385\*8,1) ...

reshape(spectrogram(m(Fs\*65:2:Fs\*70+1,1)),16385\*8,1)];

testRock = abs(testRock);

testRock = testRock - mean(testRock);

transformedRock = (Transform \* testRock)';

correctBayes = correctBayes + sum(nb.predict(transformedRock)=="rock3");

correctLda = correctLda + sum(classify(transformedRock,xtrain,ctrain)=="rock3");

% Repeat for an unused Jazz song

[m,Fs] = audioread('.\Music\Test\Jazz.mp3');

testJazz = [reshape(spectrogram(m(1:2:Fs\*5+1,1)),16385\*8,1) ...

reshape(spectrogram(m(Fs\*15:2:Fs\*20+1,1)),16385\*8,1) ...

reshape(spectrogram(m(Fs\*30:2:Fs\*35+1,1)),16385\*8,1) ...

reshape(spectrogram(m(Fs\*55:2:Fs\*60+1,1)),16385\*8,1) ...

reshape(spectrogram(m(Fs\*65:2:Fs\*70+1,1)),16385\*8,1)];

testJazz = abs(testJazz);

testJazz = testJazz - mean(testJazz);

transformedJazz = (Transform \* testJazz)';

correctBayes = correctBayes + sum(nb.predict(transformedJazz)=="jazz");

correctLda = correctLda + sum(classify(transformedJazz,xtrain,ctrain)=="jazz");

% Repeat for an unused Ambient song

[m,Fs] = audioread('.\Music\Test\Ambient.mp3');

testAmbient = [reshape(spectrogram(m(1:2:Fs\*5+1,1)),16385\*8,1) ...

reshape(spectrogram(m(Fs\*15:2:Fs\*20+1,1)),16385\*8,1) ...

reshape(spectrogram(m(Fs\*30:2:Fs\*35+1,1)),16385\*8,1) ...

reshape(spectrogram(m(Fs\*55:2:Fs\*60+1,1)),16385\*8,1) ...

reshape(spectrogram(m(Fs\*65:2:Fs\*70+1,1)),16385\*8,1)];

testAmbient = abs(testAmbient);

testAmbient = testAmbient - mean(testAmbient);

transformedAmbient = (Transform \* testAmbient)';

correctBayes = correctBayes + sum(nb.predict(transformedAmbient)=="ambient");

correctLda = correctLda + sum(classify(transformedAmbient,xtrain,ctrain)=="ambient");

BayesAccuracy = correctBayes / 25

LDAAccuracy = correctLda / 25