Bringing it Back to the Classics: SVD and Classification Algorithms

Timothy Lu

Abstract

Singular Value Decomposition is investigated in terms of effectiveness with varying number of modes. This is observed through image reconstruction. SVD is then applied toward a training set of music data. This training set is used for classification algorithms such as a Naive Bayes classifier, Linear Discriminant Analysis (LDA), and Support Vector Machines (SVM). These algorithms are measured against each other through the classification of music from different genres and different artists using tests sets from the same artists given in the training sets for data.

1. Introduction and Overview

Classification is something that we, as people, do naturally. After being told what something is, whether an object like a cup or a bowl, an animal like a cat or a dog, or a genre of music like hip hop or folk, our brains will naturally classify those things by their features to fit under a given label. This report tackles how through learning algorithms and linear algebra, machines can also classify things to some degree of accuracy.

In addition to classification, we will first conduct an overview of image recognition using Singular Value Decomposition (SVD) and the Yale faces dataset. This will be an exploration of image reconstruction using varying numbers of modes. We will also examine how cropped images with conveniently centered data (features) differ from uncropped images as well as the interpretation of the SVD on an image. This is crucial to the second step as the SVD is used to reduce dimensionality and hasten runtime efficiency by using useful points of data.

2. Theoretical Background

As SVD and PCA is outlined in the previous homework assignment, there will be limited background regarding it here. If you are seeking information about SVD or PCA outside of the scope of what is mentioned in this section, then please refer to my report for homework 3.

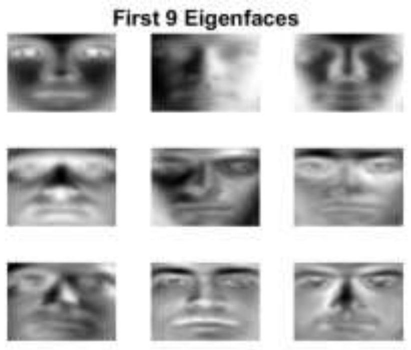

Figure 1 - First 9 Eigenfaces of a Yale Face

SVD, in reducing the dimensionality of a given data set, gives us *principal components* or features that represent our data. Each mode, then, represents a different set of features of a data set that accounts for a percentage of the total variation. As shown in figure 1, these can be literal features of the face, from mouth, eyes, complexion (or shading of the image) etc. As in the previous report, Large data of a thing can be represented in a fraction of the dimensions if using SVD. This is made possible by the important principal components. Which account for most of the variation within a given image.

"Machine Learning is based upon optimization techniques for data" (Kutz, 178). Moreover, machine learning is used to extract information or "meaningful features from data" (Kutz, 178). When used to make predictions, it is called *predictive analytics*. Machine learning can be separated into two categories, *supervised* and *unsupervised* machine learning. In *supervised* learning, the machine is given already labeled data, whereas in *unsupervised* learning, no labels are given, "thus, they must find patterns in the data in a principled way in order to determine how to cluster data and generate labels for predicting and classifying new data" (Kutz, 179).

The *Naive Bayes* algorithm is simple to construct and intuitive. As a form of supervised learning, labels are fed to the algorithm, and the algorithm classifies said data. This algorithm is based upon Bayes's theorem and conditional probabilities, "thus one can estimate the label of a new data point based on the prior probability distributions of the labeled data."

Linear Discriminant Analysis (LDA) was developed by Fisher in 1936 and is still used broadly today. LDA is another supervised learning technique with a goal "to find a linear combination of features that characterizes or separates two or more classes of objects or events in the data" (Kutz, 203). The goal of LDA then is to "find a suitable projection that maximizes the distance between the inter-class data while minimizing the intra-class data." For a two class LDA, we have the following projection w.

$$w = \arg\max_{w} \frac{w^T S_B w}{w^T S_W w} \tag{1}$$

where the scatter matrices for between-class S_B and within-class S_W data are given by

$$S_B = (\mu_2 - \mu_1)(\mu_2 - \mu_1)^T \tag{2}$$

$$S_W = \sum_{j=1}^{2} \sum_{x} (x - \mu_j)(x - \mu_j)^T \tag{3}$$

Support vector machine (SVM) is a core machine learning tool that is also widely used and often providing results better than other methods. Although with big data sets, SVM can be replaced with deep neural nets, SVM and *random forest* are some of the best performing machine learning tools. SVM optimizes is to both optimize a decision line (in the case of linear

SVM) which makes the fewest labeling errors for the data, but also the largest margin between the data. Linear SVM can be represented as

$$w \cdot x + b = 0 \tag{4}$$

where vector w and constant b parametrize the hyperplane. Linear classifiers are of limited value, however, as they are far too restrictive for different data, especially data occupying high-dimensional space and may not be structured into two groups separable by hyperplane. This is accomplished by mapping the data into a nonlinear, higher dimensional space

$$x \mapsto \Phi(x) \tag{5}$$

where $\Phi(x)$ is thus new *observables* of the data. Thus, SVM learns the "hyperplanes that optimally split the data into distinct clusters in a new space" (211, Kutz). Now the hyperplane function is

$$f(x) = w \cdot \Phi(x) + b \tag{6}$$

With corresponding labels $y_j \in \{\pm 1\}$ for each point $f(x_j)$. SVM is a powerful tool for accurately classifying data.

3. Algorithm Implementation and Development

All of the code for the following algorithms is outlined in Appendix B.

3.1. Image Reconstruction

1. Read in images and reshape them to column vectors to place in an image matrix. Convert each vector to a double.
2. Take the SVD of the given image matrix and plot the energies associated with the modes.
3. Select a number of features/modes (a range of columns in the Σ matrix) to use for image reconstruction.
4. Multiply $U\Sigma V^T$ to get a reconstruction of the original image matrix.

3.2. Music Classification

1. Read in audio data and downsample the data for efficiency (such that each clip does not take up too much space) and convert it to mono.
2. Cut five second clips of the data.
3. Take the Fourier Transform of each five second clip such that we have time and frequency data.
    a. Filter each time window using a gaussian filter.
    b. Take the absolute data of the frequency.
4. Construct a matrix with each column as a vector representing a five second clip, organized by artist/genre.
5. Perform the SVD on this larger matrix with all songs as column vectors. The resulting matrix V is dimensionally lower, thus making it easier to use in machine learning.
6. Randomly choose 270 snippets to comprise a training set from each "class" of music. The remaining 30 snippets are to be used as a test set for our classifiers.
7. Choose the number of modes (100) that will be used. This is represented as columns of matrix V whereas rows of V are song snippets.

8. Stack up the snippets as well as the modes chosen into a single matrix for the training set. Do the same for the test set.
9. Construct a matrix with the “answers” or correct classifications for the test set (a number 1, 2, or 3).
10. Use the corresponding MATLAB function for Naïve Bates, LDA, or SVM to get a prediction for the test sets based on the training sets.
11. Repeat this process 100 times to cross validate and adjust for randomness.

4. Computational Results

4.1. Image Reconstruction

As shown in figure 1, the eigenfaces represent key features of the images. Each mode, or principal component then, represents different features of the original image that we can use to recreate a given image. Each face in figure 1 highlight a prominent feature of the face, from shading to the outlines of nose, eyes, and mouth.

Figure 2 represents both the singular values as well as a reconstruction of an uncropped image from the Yale database. The image on the right is the plot of singular values as well as the energy corresponding to each mode. As we can see, with all the cropped images together, it takes many modes to accurately represent the data.

The most prominent principal component only represents roughly 12% of the data. Thus, we must use many nodes, almost full rank to get a clear reconstruction. This is represented in the image on the left of figure 2. There are 11 modes total (corresponding to 11 images of the subject), and we see that using 1 mode is far too blurry for reconstruction (doesn’t represent the variation enough). While using 4 modes is also blurry, as we near the full rank (10 modes), we have a more and more clear picture. Thus, for reconstruction of an image, we must account for a significant amount of the variation using principal components. Although the SVD reduces dimensionality, SVD is less helpful when the principal components do not account for enough of the variation.

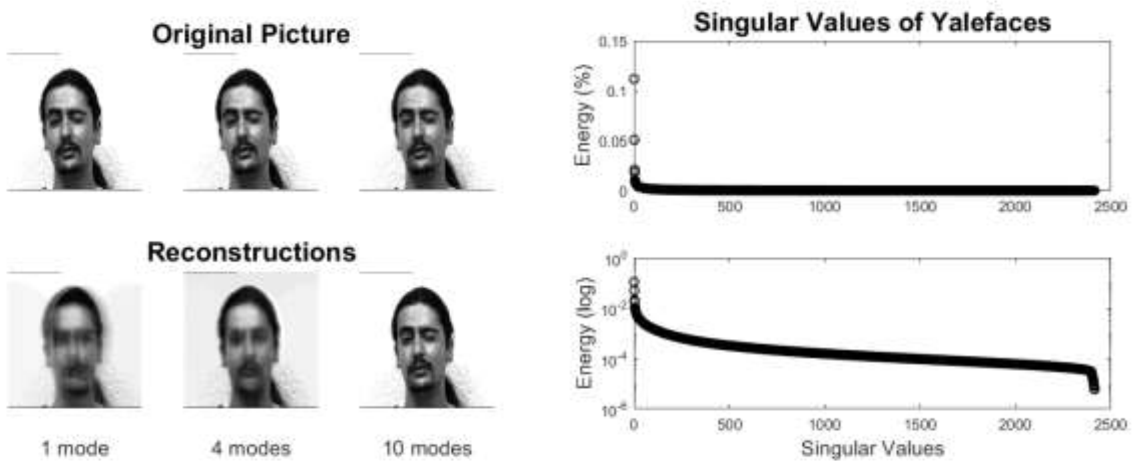

Figure 2 - Reconstruction of an image given varying modes (left). Singular value plot with corresponding energies (right)

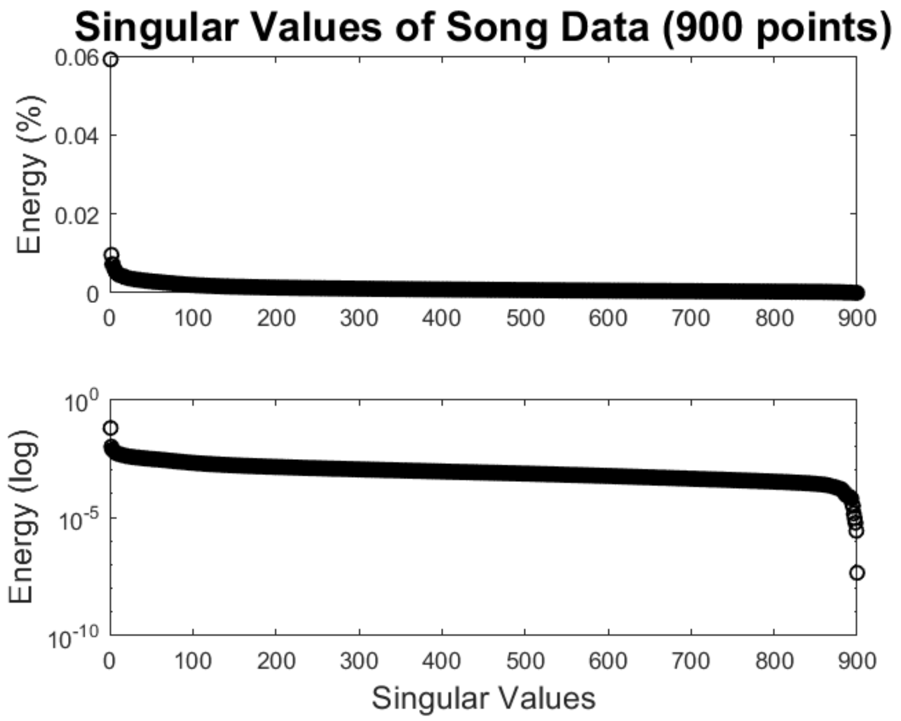

Figure 3 - Singular value plot of all songs used for training & testing (test case 1)

4.2. Music Classification

We first take a look at singular values for the data we are training and testing on and given that much of the variation lies within the first roughly 100 modes, we use 100 modes for our reconstruction of the data through SVD.

We have three separate test cases when it comes to music classification:

1. We try and classify songs from three different artists in three separate genres.
2. Classify songs from three different artists in the same genre.
3. Classify song from various artists in three different genres, three for each genre.

The accuracy for each classifier in given in the following table:

	Naive Bayes		**LDA**		**SVM**	
	Mean (%)	std	Mean (%)	Std	Mean (%)	Std
Case 1	0.6199	0.0381	0.914	0.0282	0.914	0.0265
Case 2	0.8611	0.0358	0.9411	0.0219	0.9414	0.0214
Case 3	0.6587	0.046	0.7443	0.0435	0.7466	0.0399

4.2.1. Test Case 1: Three Genres, Three Artists

We have data from three albums: *Illmatic*, by Nas representing 90s rap, *Rocky Mountain High*, by John Denver representing 70s folk, and *Innervisions*, by Stevie Wonder representing 70s soul/R&B. When put through the classifiers, we see in the table above that while LDA and SVM

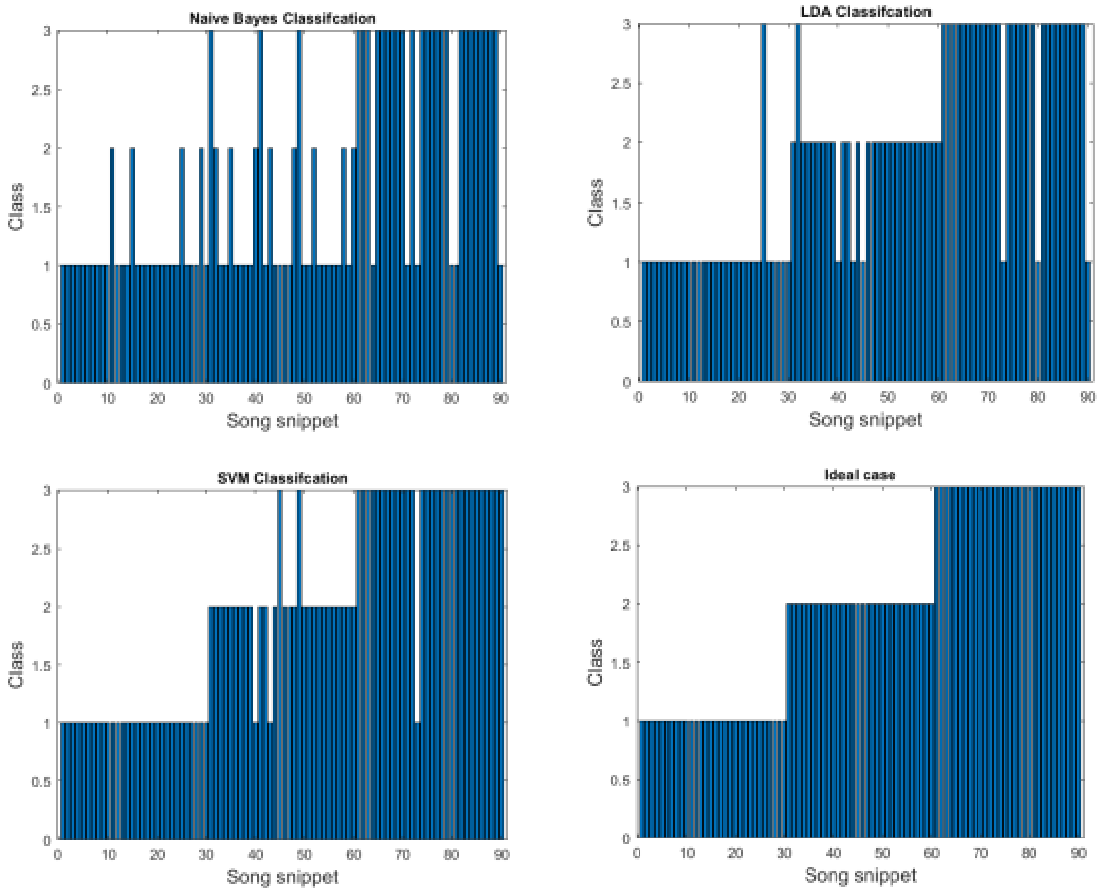

Figure 4 - One run of results of classifying using three different classifiers versus the ideal case

are fairly accurate with 91.4% classification accuracy, Naive Bayes is quite a bit lower with a mean of 61.9%. An example of a run of the classifiers is given in figure 4 compared to the ideal case. We see that for test case 1, Naive Bayes is much less accurate than the other two classifiers.

4.2.2. Test Case 2: Three Artists, One Genre

For case 2 we tested the ablums *Midnight Mauraders*, by A Tribe Called Quest, *Illmatic*, by Nas, and *Enter the Wu-Tang (36 Chambers)*, by Wu-Tang Clan.

An intuition for case 2 might be that the classifiers might have a harder time distinguishing from artists of the same genre than artists of different genres. However, when looking at the data this is hardly the case. In fact, all of our classifiers performed better in case 2 as opposed to case 1. Naive Bayes, while still trailing LDA and SVM at $\cong$ 94%, shoots up to $\cong$ 86%. While the other two remain at the same level. This is particularly noteworthy given the discrepancy between the classifiers in the previous test case. The variation here is overall much less than case 1. This might be because given similarities in the same genre of music, there are more prominent features that differ. In other words, more of the variation might fall under less of the features.

This data is plotted in figure 5. We see here that compared to test case 1, the performance of the classifiers is much better, and much closer to the ideal situation (or the answer).

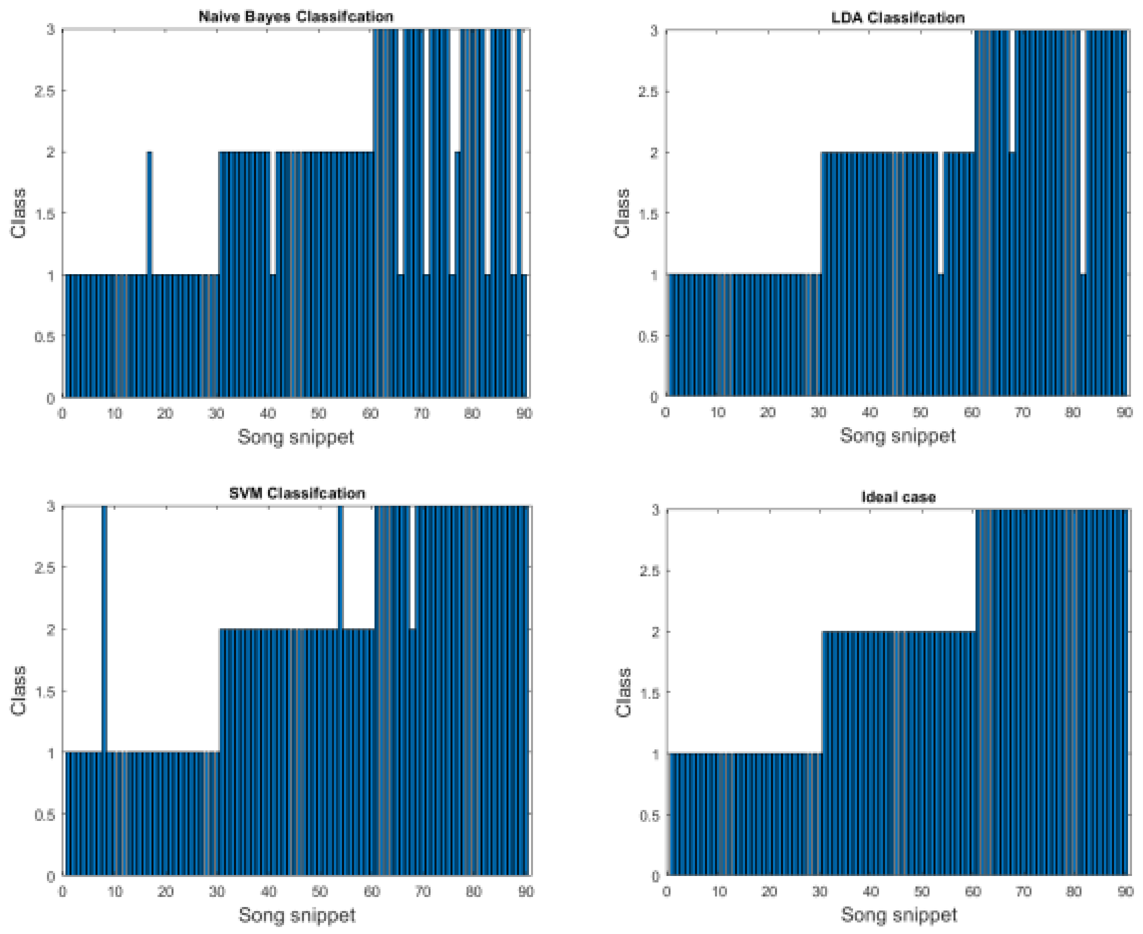

Figure 5 - Naive Bayes (top left), LDA (top right), SVM (bottom left), Ideal (bottom right) for one run in case 2

4.2.3. Test Case 3: Three Genres, Nine Artists

The albums used here were: *Angel Clare*, by Art Garfunkel, *Rocky Mountain High*, by John Denver, and *Don Quixote*, by Gordon Lightfoot representing 70s folk; *Brother, Brother, Brother,* by The Isley Brothers, *Innervisions,* by Stevie Wonder, and *Trouble Man,* by Marvin Gaye to represent 70s Soul/R&B; and *Midnight Mauraders,* by A Tribe Called Quest, *Enter the Wu-Tang (36 Chambers),* by Wu-Tang Clan, and *Illmatic,* by Nas.

One run of this data is represented in figure 6. We see in our summary statistics that there is significantly worse performance by every classifier for this case in comparison to case 2, while only the Naive Bayes performs better in classification in comparison to case 1. This means that classifiers were performing worse in predictions given the differing artist and differing genres. This could imply that with so many artists and so many genres, there are more features that account for variation, and thus more modes to be used when applying SVD to our data prior to training and testing.

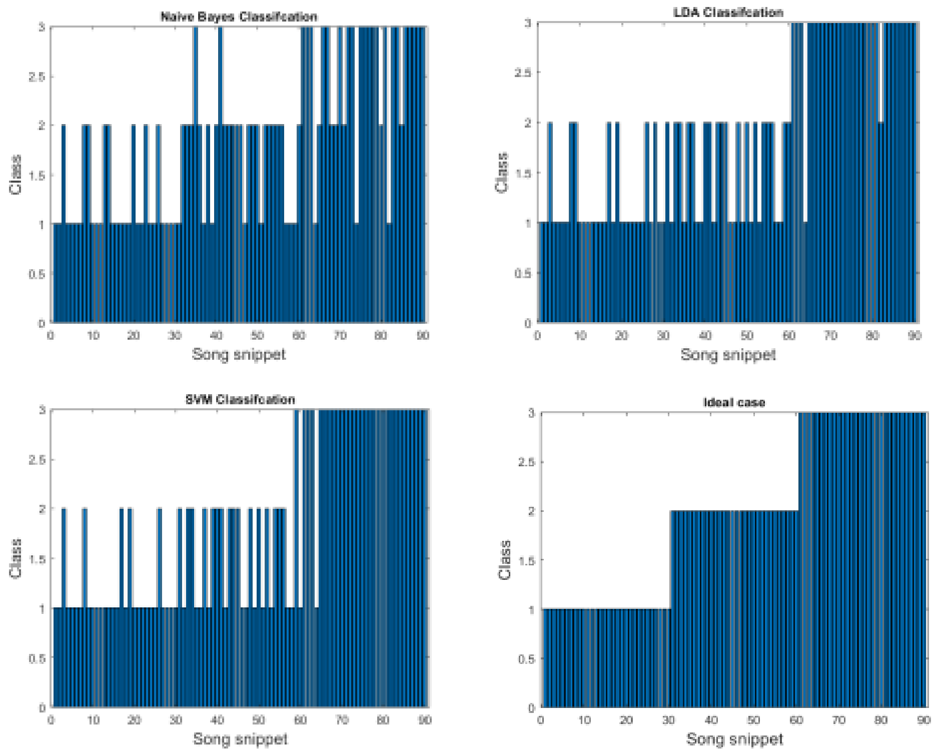

Figure 6 - Naive Bayes (top left), LDA (top right), SVM (bottom left), Ideal (bottom right) for one run in case 3

5. Summary and Conclusions

What we see here is that SVD lowers dimensionality, but also lower the accuracy of various algorithms that use the data output from SVD, given that we use less modes (features) in a given algorithm. With image reconstruction, images became more blurry upon reconstruction when using less nodes (less features) whereas images became more clear when given more features to work with. The columns in the **U** matrix are orthonormal basis vectors for the "facespace" while **Σ** is a diagonal matrix of eigenvalues/features of **A**, and **V** tells us how the image is projected onto **U** and **Σ**.

In a similar fashion, we saw this illustrated with our classifiers as well. In Test case 1, there were a moderate number of features that separated the snippets of song from one another. With only three bands, however, there were not nearly as many features separating songs as in test case 3. This is why in test case 3, there was lower performance in classifiers as the number of modes used remained the same while the number of relevant modes increased. In test case 2, because the songs were of the same genre/era, the number of relevant modes decreased. For example, while *Illmatic* and *Enter the Chamber* are very different from a hip-hop fan's point of view, they use similar instrumentation (or sampling techniques and drum beats) with slight differences in style or voice. In contrast, there is much more variation between *John Denver*'s music and *Nas's* music in instrumentation, melodies, voice etc. Thus, because there are more differences, there are more relevant features that make up those differences.

We see from these experiments that SVD is very powerful and useful, especially in classification algorithms. However, while SVD may be very useful, one must consider the number of modes/dimensions needed to accurately perform an algorithm.

Appendix A

Command	Implementation
fullfile	Builds full file name from a directory and name. Used to read in data.
svd	Used to perform the SVD on a given matrix. Usually used 'econ' as an option.
reshape	Used to reshape the given matrix to different dimensions. Handled turning images into column vectors.
diag, sum	"diag" gives the diagonal of a given matrix. Sum sums a vector together. Used to find energies of singular values.
imagesc, colormap	Used to plot the images from Yale faces.
audioread	Used to read in audio data (.mp3 data).
downsample	Used to sample audio data down into a digestible amount.
fft	Performs the Fast Fourier Transform on data. Used to get frequency and time data from audio files.
size	Gives the size of a matrix
fitcnb, nb.predict	Creates a Naïve Bayes classifier for a given training set and corresponding labels. Used to predict.
classify	Uses LDA to classify test data using training data and corresponding labels.
fitcecoc, predict	Creates a SVM classifier for a given training set and its corresponding labels. Used to predict with more than two classes.

Appendix B

```
clear all; close all; clc

cropped = 0;

if cropped == 1
    data_fold = 'C:\Users\timot\Desktop\amath 
482\hw4\CroppedYale\yaleB';
    pgm_files = [];
    for j = 1:39
        if j < 10
            data_folder = strcat(data_fold, '0', num2str(j),'\');
        else
            data_folder = strcat(data_fold, num2str(j), '\');
        end
        file_extension = fullfile(data_folder, '*.pgm');
        pgm_files = [pgm_files; dir(file_extension)];
    end
else
    data_fold = 'C:\Users\timot\Desktop\amath 482\hw4\yalefaces\';
    pgm_files = [];
    for j = 9:9
        if j < 10
            file_extension = fullfile(data_fold, strcat('subject0', 
num2str(j), '.*'));
        else
            file_extension = fullfile(data_fold, strcat('subject', 
num2str(j), '.*'));
        end
        pgm_files = [pgm_files; dir(file_extension)];
    end
end

image_matrix = [];

for j = 1:length(pgm_files)
    baseName = pgm_files(j).name;
    if cropped == 1
        data_folder = strcat(data_fold, baseName(6:7));
    else
        data_folder = data_fold;
    end

    fullName = fullfile(data_folder, baseName);
    fprintf(1, 'Now reading %s\n', fullName);
    image_matrix2 = imread(fullName);

    image_matrix_reshaped = reshape(image_matrix2, [], 1);
    image_matrix(:,j) = double(image_matrix_reshaped);
end
```

```matlab
%% SVD

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


[m,n] = size(image_matrix);
[IMAGEM, IMAGEN] = size(image_matrix2);

% for image reconstruction
S_rank9 = S;
S_rank9(2:end,2:end) = 0;
A = reshape(image_matrix(:,n),IMAGEM, IMAGEN);
A_9 = U*S_rank9*V';
S_rank50 = S;
S_rank50(5:end,5:end) = 0;
A_50 = U*S_rank50*V';
S_rank100 = S;
S_rank100(11:end,11:end) = 0;
A_100 = U*S_rank100*V';

%% Singular values plot
sig = diag(S)/sum(diag(S));
subplot(2,1,1), plot(sig,'ko','Linewidth',[1.1])
ylabel('Energy (%)','Fontsize',14)
title('Singular Values of Yalefaces', 'Fontsize', 18);
subplot(2,1,2), semilogy(sig,'ko','Linewidth',[1.1])
ylabel('Energy (log)','Fontsize',14)
xlabel('Singular Values', 'Fontsize',14);
```

```matlab
%% Eigenfaces plot
for j=1:9
    ef = reshape(U(:,j),IMAGEM,IMAGEN);
    subplot(3,3,j)
    imagesc(ef), colormap(gray), axis off
    if j == 2
        title('First 9 Eigenfaces','Fontsize',18);
    end
end

%% Reconstruction
[m,n] = size(image_matrix);
ave_9 = zeros(m,1);
ave_50 = zeros(m,1);
ave_100 = zeros(m,1);
for j=1:n
    ave_100 = ave_100 + A_100(:,j)/n;
    ave_50 = ave_50 + A_50(:,j)/n;
    ave_9 = ave_9 + A_9(:,j)/n;
end
face_100 = reshape(ave_100,IMAGEM,IMAGEN);
face_50 = reshape(ave_50,IMAGEM,IMAGEN);
face_9 = reshape(ave_9,IMAGEM,IMAGEN);

subplot(1,3,3)
imagesc(face_100), colormap(gray), axis off
subplot(1,3,2)
imagesc(face_50), colormap(gray), axis off
title("Average yaleface 9, 50, 100 modes",'Fontsize',18)
subplot(1,3,1)
imagesc(face_9), colormap(gray), axis off
subplot(2,3,1)
imagesc(A), colormap(gray),axis off
subplot(2,3,2)
imagesc(A), colormap(gray),axis off
title("Original Picture",'Fontsize',18)
subplot(2,3,3)
imagesc(A), colormap(gray),axis off

subplot(2,3,6)
imagesc(reshape(A_100(:,n),IMAGEM, IMAGEN)), colormap(gray), axis off
xlabel('10 modes','Fontsize',14)
set(findall(gca, 'type', 'text'), 'visible', 'on')
subplot(2,3,5)
imagesc(reshape(A_50(:,n),IMAGEM, IMAGEN)), colormap(gray), axis off
xlabel('4 modes','Fontsize',14)
set(findall(gca, 'type', 'text'), 'visible', 'on')
title("Reconstructions",'Fontsize',18)
subplot(2,3,4)
imagesc(reshape(A_9(:,n),IMAGEM, IMAGEN)), colormap(gray), axis off
xlabel('1 mode','Fontsize',14)
set(findall(gca, 'type', 'text'), 'visible', 'on')
```

Music Genre Classification

```
clear all; close all; clc

% get data
cd('C:\Users\timot\Desktop\amath 482\hw4\music\')
artistid = 'nas';
files = dir(strcat(artistid,'*.mp3'));
samples = 300;
rf = 100; %scale by which we reduce our sample
nas = []

for file = files'
    [y, Fs] = audioread(file.name);
    z = downsample(y, rf);
    z2 = z(:,1);
    z3 = z2';

    fivesec = round(5*(Fs/rf));

    for j=0:1:length(z3)/fivesec
        if (j+1)* fivesec < length(z3) % get five second intervals
            a= z3(j * fivesec + 1:(j+1)* fivesec);
            L=5;
            n = length(a);
            t2 = linspace(0,L,n+1);
            t=t2(1:n);
            k = (2.0*pi/L)*[0:n/2-1 -n/2:-1];
            ks = fftshift(k);
            Sgt_spec=[];
            tslide = 0:0.1:5;
            for j=1:length(tslide)
                g = exp(-2.5*(t - tslide(j)).^2);
                Sg = g.*a;
                Sgt = fft(Sg);
                Sgt_spec=[Sgt_spec; abs(fftshift(Sgt))];
            end
            [m n] = size(Sgt_spec);
            nas = [nas reshape(Sgt_spec,m*n, 1)];
            [filler snippet_no] = size(nas);
            snippet_no
        end
        if snippet_no >= samples
            break
        end
    end
    if snippet_no >= samples
            break
    end
end
save(strcat(artistid, '.mat'),artistid);
```

```
close all; clear all; clc
load jd.mat; % John Denver, Folk
load nas.mat; % Nas, 90s Rap
load sw.mat; % Stevie Wonder, Soul
load tcq.mat; % A Tribe Called Quest, 90s Rap
load wtc.mat; % Wu Tang Clan, 90s Rap

%%

[mm nn] = size(nas);

X = [tcq wtc nas];

% SVD
[u,s,v] = svd(X, 'econ');
%%

sig = diag(s)/sum(diag(s));
subplot(2,1,1), plot(sig,'ko','Linewidth',[1.1])
ylabel('Energy (%)','Fontsize',14)
title('Singular Values of Song Data (900 points)', 'Fontsize', 18);
subplot(2,1,2), semilogy(sig,'ko','Linewidth',[1.1])
ylabel('Energy (log)','Fontsize',14)
xlabel('Singular Values', 'Fontsize',14);
```

```
%% classification
clc
bestpct = [0 0]; best = [1 1;1 1];
pctLDA = zeros(100,1); pctNB = zeros(100,1); pctSVM = zeros(100,1);
for j = 1:100
    trsize = 270;
    testsize = nn-trsize;
    features = 100;

    q1 = randperm(nn); q2 = randperm(nn); q3 = randperm(nn);

    adata = v(1:nn, 1:1+features);
    bdata = v(nn+1:nn*2, 1:1+features);
    cdata = v(nn*2+1:nn*3, 1:1+features);
    xtrain=[adata(q1(1:trsize),:);
        bdata(q2(1:trsize),:);
        cdata(q3(1:trsize),:)];
    xtest=[adata(q1(trsize+1:end),:);
        bdata(q2(trsize+1:end),:);
        cdata(q3(trsize+1:end),:)];
    ctrain=[ones(trsize,1); 2*ones(trsize,1); 3*ones(trsize,1)];

    nb=fitcnb(xtrain,ctrain);
    prenb=nb.predict(xtest);
    preld=classify(xtest,xtrain,ctrain);
    svm=fitcecoc(xtrain,ctrain);
    presvm = predict(svm,xtest);

    figure(1)
    bar(prenb);
    figure(2)
    bar(preld);
    figure(3)
    bar(presvm);

    answer = [ones(testsize,1) 2*ones(testsize,1) 3*ones(testsize,1)];
    countLDA = 0; countNB = 0; countSVM = 0;
    for k = 1:90
        if prenb(k) == answer(k)
            countNB = countNB + 1;
        end
        if preld(k) == answer(k)
            countLDA = countLDA + 1;
        end
        if presvm(k) == answer(k)
            countSVM = countSVM + 1;
        end
    end
    pctLDA(j)=countLDA/90; pctNB(j)=countNB/90; pctSVM(j)=countSVM/90;
end
mean(pctNB) std(pctNB) mean(pctLDA) std(pctLDA) mean(pctSVM)
std(pctSVM)
```