Jeremy Aguirre

AMATH 482

March 6, 2020

Assignment #4: Methods for Music Classification

Abstract

In this assignment I explored different music classification techniques and tested them in different scenarios. I applied the k-nearest numbers, naïve Bayes, and linear discriminant analysis classification methods to the classification of songs from bands of different genres, bands of the same genre, and different genres. After 1,000 trials of each method in each scenario, I found that the k-nearest neighbors method was the most consistently accurate.

I. Introduction and Overview

I acquired music from various bands in order to test different classification methods. I applied the k-nearest numbers, naïve Bayes, and linear discriminant analysis methods. I used them to classify songs from bands of different genres, bands of the same genre, and different genres as three tests.

II. Theoretical Background

Singular Value Decomposition

The representation of the full SVD is as follows:

$$\mathbf{A} = \mathbf{U}\mathbf{\Sigma}\mathbf{V}^{*} \tag{1}$$

$$\mathbf{U} \in \mathbb{C}^{m\times m} \text{ is unitary}$$

$$\mathbf{V} \in \mathbb{C}^{n\times n} \text{ is unitary}$$

$$\mathbf{\Sigma} \in \mathbb{R}^{m\times n} \text{ is diagonal}$$

This decomposition tool is used to isolate constitutive elements from the matrix **A** which can be interpreted based on the application. A common example used to describe the SVD is the decomposition of a disk characterized by a 2x2 matrix. In this example, $\mathbf{V}^{*}$ is thought of as the component which rotates a unit disk. $\mathbf{\Sigma}$ is responsible for the stretching or compression of the disk for this case in two dimensions. The values along the diagonal of $\mathbf{\Sigma}$ are known as the singular values (σ_i). The matrix **U** then rotates the now ellipse again to yield an ellipse characterized by matrix **A.** The singular vale decomposition has various uses depending on the application.

Supervised Learning Algorithms

k-Nearest Neighbors

k-Nearest neighbors classification is a non-parametric method which classifies an object based on the classification of its neighbors. For example, with a k-value of three, a test data point will be labeled based on the most common label among its three nearest neighbors among the training data.

Naïve Bayes

Naïve Bayes refers to conditional probabilistic classifiers which apply Bayes' theorem:

$$p(x|y) = \frac{p(y|x)p(x)}{p(y)} \quad (2)$$

This method uses the probability distributions of labeled training data to predict the label of given test data.

Linear Discriminant Analysis

Linear Discrimination Analysis is used as a dimensional reduction technique in linear classification. It projects data onto a lower-dimensional space in a way that provides the most separation between inter-class data while grouping intra-class data.

III. Algorithm Implementation and Development

Test 1: Band Classification

I acquired multiple songs from the bands Pierce the Veil, Bach, and Johnny Cash to produce data for the first test. I used the command audioread to read each music file and produce sample data. This produced two sets of data corresponding to a right channel and left channel, so I had to combine the data. I also reduced the data size to 20,000 points using the command resample. I then extracted 25, five-second intervals of data from each song and used the command spectrogram to extract frequency information from the songs. Singular value decomposition was then applied to the normalized frequency data using the command svd. The resulting matrix **V** from equation 1 contains information of each five-second interval for each mode. By plotting different modes and their corresponding information, I determined that mode 3, 4, and 5 showed the best separation between classes. For the implementation of the k-nearest neighbors, naïve Bayes, and linear discriminant analysis I used data from the 3rd, 4th, and 5th modes. I used the command randperm to randomly generate numbers between 1 and 50 to use different data for the training set and test set for each trial of the classification. I chose a 2:3 ratio for the test set to training set sizes. I modified the output of each algorithm by assigning labels to the indices of the data corresponding to each band. I then calculated the accuracy of the label for each output and ran each of the three classifiers 1,000 times for cross validation. The average accuracy of each of the three method was then recorded for comparison.

Test 2: The Case for Seattle

For the second test, I acquired multiple songs from the bands Soundgarden, Alice In Chains, and Pearl Jam to produce data. The implementation of k-nearest neighbors, naïve Bayes, and linear discriminant analysis was the same as described in test 1.

Test 3: Genre Classification

For the third test, I acquired multiple songs from the genres of classical, country and rock to produce data. I used songs from the bands Pierce the Veil, AC/DC, Judas Priest, Johnny Cash, Conway Twitty, Waylon Jennings, Bach, Beethoven, and Mozart. The implementation of k-nearest neighbors, naïve Bayes, and linear discriminant analysis was the same as described in test 1.

IV. Computational Results

Test 1: Band Classification

After 1,000 trials, k-nearest neighbors performed with 98.42% accuracy, naïve Bayes had an accuracy of 96.83%, and linear discriminant analysis was 95.31% accurate when classifying bands of different genres.

Test 2: The Case for Seattle

After 1,000 trials, k-nearest neighbors performed with 95.69% accuracy, naïve Bayes had an accuracy of 92.11%, and linear discriminant analysis was 93.45% accurate when classifying bands of the same genre.

Test 3: Genre Classification

After 1,000 trials, k-nearest neighbors performed with 96.76% accuracy, naïve Bayes had an accuracy of 92.93%, and linear discriminant analysis was 89.25% accurate when classifying bands of different genres.

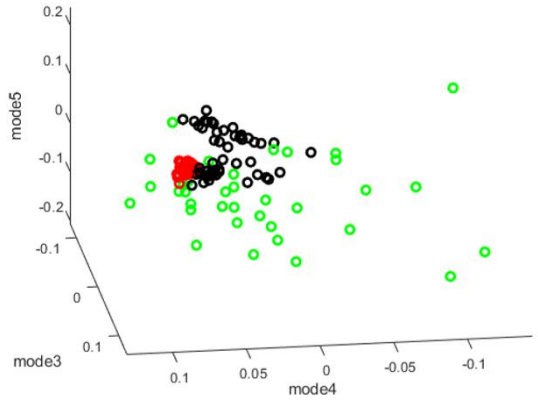

Figure 1. Plot of class separation used to choose modes for implementation of test 1. It shows the data corresponding to the chosen modes 3, 4 and 5. Black corresponds to Pierce the Veil, Red corresponds to Bach, and Green corresponds to Johnny Cash.

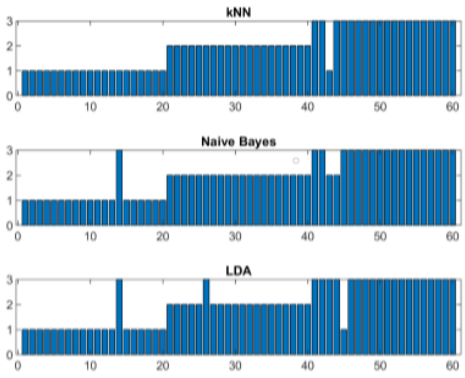

Figure 2. Display of results from the k-nearest neighbors, naïve Bayes, and linear discriminant analysis for the final iteration of test 1.

Tests	k-Nearest Neighbors	Naïve Bayes	Linear Discriminant Analysis
Band Classification	98.42%	96.83%	95.31%
The Case for Seattle	95.69%	92.11%	93.45%
Genre Classification	96.76%	92.93%	89.25%

Table 1. The average accuracy of each classification method in Test 1, 2, and 3 over 1,000 trials.

V. Summary and Conclusions

In this assignment I tested the efficacy of three different methods for music classification. I applied the k-nearest numbers, naïve Bayes, and linear discriminant analysis classification methods. I applied them to the classification of songs from bands of different genres, bands of the same genre, and different genres (Test 1, 2, and 3, respectively). Ultimately, I found that the k-nearest neighbors method was the most consistently accurate. The linear discriminant analysis and naïve Bayes methods experienced a drop-off in performance distinguishing between more similar classes (Test 2 and 3). I also noticed significant drops in accuracy for all methods when decreasing the number trials, stressing the need for extensive cross-validation when using such methods.

Appendix A

[u, s, v] = svd(x) - MATLAB function to perform the singular value decomposition

Appendix B

```
%% AMATH 482: Assignment #4 - Test 1
path = 'c:\Users\jerem\OneDrive\Desktop\AMATH 482\Music1';
file = dir(path);
data = [];
for i = 3:length(file)
   filename = file(i).name;
   [song, Fs] = audioread(strcat(path, '/', filename));
   song = song(:,1) + song(:,2);
   song = resample(song, 20000, Fs);
   Fs = 20000;
   for j = 1:5:125
      test = song(Fs*j:Fs*(j+5), 1);
      spec_test = abs(spectrogram(test));
      spec_test = reshape(spec_test, 1, 8*32769);
      data = [data spec_test'];
   end
end
```

```
%% Plotting Dominant Spectrogram Modes
[u, s, v] = svd(data - mean(data(:)), 'econ');
plot(diag(s) ./ sum(diag(s)), 'ko');
plot3(v(1:50, 3), v(1:50, 4), v(1:50, 5), 'ko'), hold on
plot3(v(51:100, 3), v(51:100, 4), v(51:100, 5), 'ro')
plot3(v(101:150, 3), v(101:150, 4), v(101:150, 5), 'go')
xlabel('Mode 3')
ylabel('Mode 4')
zlabel('Mode 5')

%% k-Nearest Neighbors
result_knn = [];
for i = 1:1000
   q1 = randperm(50);
   q2 = randperm(50);
   q3 = randperm(50);
   xptv = v(1:50, 3:5);
   xbach = v(51:100, 3:5);
   xjc = v(101:150, 3:5);
   xtrain = [xptv(q1(1:30), :) xbach(q2(1:30), :) xjc(q3(1:30),:)];
   xtest = [xptv(q1(31:end), :) xbach(q2(31:end), :) xjc(q3(31:end),:)];
   ctrain = [ones(30,1) 2*ones(30,1) 3*ones(30,1)];
   ctest = [ones(20,1) 2*ones(20,1) 3*ones(20,1)];

   index = knnsearch(xtrain, xtest, 'A', 3);
   pre = [];
   true = 0;
   for j = 1:length(index)
      ind = index(j,:);
      x0 = [ctrain(ind(1)) ctrain(ind(2)) ctrain(ind(3))];
      x0 = mode(x0);
      pre = [pre x0];
      if pre == ctest(j,1)
         true = true + 1;
      end
   end
   acc = true/length(index);
   subplot(3,1,1)
   bar(pre);
   title('kNN')
end
result_knn = mean(acc);
```

```
%% Naive Bayes
result_nb = [];
for i = 1:1000
   q1 = randperm(50);
   q2 = randperm(50);
   q3 = randperm(50);
   xptv = v(1:50, 3:5);
   xbach = v(51:100, 3:5);
   xjc = v(101:150, 3:5);
   xtrain = [xptv(q1(1:30), :) xbach(q2(1:30), :) xjc(q3(1:30),:)];
   xtest = [xptv(q1(31:end), :) xbach(q2(31:end), :) xjc(q3(31:end),:)];
   ctrain = [ones(30,1) 2*ones(30,1) 3*ones(30,1)];
   ctest = [ones(20,1) 2*ones(20,1) 3*ones(20,1)];

   nb = fitcnb(xtrain,ctrain);
   pre = nb.predict(xtest);
   true = 0;
   trials = size(xtest,1);
   for j = 1:trials
      if pre(j,1) == ctest(j,1)
         true = true + 1;
      end
   end
   acc = true/trials;
   subplot(3,1,2)
   bar(pre)
   title('Naive Bayes')
end
result_nb = mean(acc);

%% Linear Discriminant Analysis
result_lda = [];
for i = 1:1000
   q1 = randperm(50);
   q2 = randperm(50);
   q3 = randperm(50);
   xptv = v(1:50, 3:5);
   xbach = v(51:100, 3:5);
   xjc = v(101:150, 3:5);
   xtrain = [xptv(q1(1:30), :) xbach(q2(1:30), :) xjc(q3(1:30),:)];
   xtest = [xptv(q1(31:end), :) xbach(q2(31:end), :) xjc(q3(31:end),:)];
   ctrain = [ones(30,1) 2*ones(30,1) 3*ones(30,1)];
   ctest = [ones(20,1) 2*ones(20,1) 3*ones(20,1)];
```

```
    pre = classify(xtest,xtrain,ctrain);
    true = 0;
    trials = size(xtest,1);
    for j = 1:trials
        if pre(j,1) == ctest(j,1)
            true = true + 1;
        end
    end
    acc = true/trials;
    subplot(3,1,3)
    bar(pre)
    title('LDA')
end
result_lda = mean(acc);


%% AMATH 482: Assignment #4 - Test 2
path = 'c:\Users\jerem\OneDrive\Desktop\AMATH 482\Music2';
file = dir(path);
data = [];
for i = 3:length(file)
    filename = file(i).name;
    [song, Fs] = audioread(strcat(path, '/', filename));
    song = song(:,1) + song(:,2);
    song = resample(song, 20000, Fs);
    Fs = 20000;
    for j = 1:5:125
        test = song(Fs*j:Fs*(j+5), 1);
        spec_test = abs(spectrogram(test));
        spec_test = reshape(spec_test, 1, 8*32769);
        data = [data spec_test'];
    end
end

%% Plotting Dominant Spectrogram Modes
[u, s, v] = svd(data - mean(data(:)), 'econ');
plot(diag(s) ./ sum(diag(s)), 'ko');
plot3(v(1:50, 3), v(1:50, 4), v(1:50, 5), 'ko'), hold on
plot3(v(51:100, 3), v(51:100, 4), v(51:100, 5), 'ro')
plot3(v(101:150, 3), v(101:150, 4), v(101:150, 5), 'go')
xlabel('Mode 3')
ylabel('Mode 4')
zlabel('Mode 5')
```

```
%% k-Nearest Neighbors
result_knn = [];
for i = 1:1000
   q1 = randperm(50);
   q2 = randperm(50);
   q3 = randperm(50);
   xptv = v(1:50, 3:5);
   xbach = v(51:100, 3:5);
   xjc = v(101:150, 3:5);
   xtrain = [xptv(q1(1:30), :) xbach(q2(1:30), :) xjc(q3(1:30),:)];
   xtest = [xptv(q1(31:end), :) xbach(q2(31:end), :) xjc(q3(31:end),:)];
   ctrain = [ones(30,1) 2*ones(30,1) 3*ones(30,1)];
   ctest = [ones(20,1) 2*ones(20,1) 3*ones(20,1)];

   index = knnsearch(xtrain, xtest, 'A', 3);
   pre = [];
   true = 0;
   for j = 1:length(index)
      ind = index(j,:);
      x0 = [ctrain(ind(1)) ctrain(ind(2)) ctrain(ind(3))];
      x0 = mode(x0);
      pre = [pre x0];
      if pre == ctest(j,1)
         true = true + 1;
      end
   end
   acc = true/length(index);
   subplot(3,1,1)
   bar(pre);
   title('kNN')
end
result_knn = mean(acc);

%% Naive Bayes
result_nb = [];
for i = 1:1000
   q1 = randperm(50);
   q2 = randperm(50);
   q3 = randperm(50);
   xptv = v(1:50, 3:5);
   xbach = v(51:100, 3:5);
   xjc = v(101:150, 3:5);
   xtrain = [xptv(q1(1:30), :) xbach(q2(1:30), :) xjc(q3(1:30),:)];
   xtest = [xptv(q1(31:end), :) xbach(q2(31:end), :) xjc(q3(31:end),:)];
```

```
    ctrain = [ones(30,1) 2*ones(30,1) 3*ones(30,1)];
    ctest = [ones(20,1) 2*ones(20,1) 3*ones(20,1)];

    nb = fitcnb(xtrain,ctrain);
    pre = nb.predict(xtest);
    true = 0;
    trials = size(xtest,1);
    for j = 1:trials
        if pre(j,1) == ctest(j,1)
            true = true + 1;
        end
    end
    acc = true/trials;
    subplot(3,1,2)
    bar(pre)
    title('Naive Bayes')
end
result_nb = mean(acc);

%% Linear Discriminant Analysis
result_lda = [];
for i = 1:1000
    q1 = randperm(50);
    q2 = randperm(50);
    q3 = randperm(50);
    xptv = v(1:50, 3:5);
    xbach = v(51:100, 3:5);
    xjc = v(101:150, 3:5);
    xtrain = [xptv(q1(1:30), :) xbach(q2(1:30), :) xjc(q3(1:30),:)];
    xtest = [xptv(q1(31:end), :) xbach(q2(31:end), :) xjc(q3(31:end),:)];
    ctrain = [ones(30,1) 2*ones(30,1) 3*ones(30,1)];
    ctest = [ones(20,1) 2*ones(20,1) 3*ones(20,1)];

    pre = classify(xtest,xtrain,ctrain);
    true = 0;
    trials = size(xtest,1);
    for j = 1:trials
        if pre(j,1) == ctest(j,1)
            true = true + 1;
        end
    end
    acc = true/trials;
    subplot(3,1,3)
    bar(pre)
```

```
    title('LDA')
end
result_lda = mean(acc);


%% AMATH 482: Assignment #4 - Test 3
path = 'c:\Users\jerem\OneDrive\Desktop\AMATH 482\Music3';
file = dir(path);
data = [];
for i = 3:length(file)
    filename = file(i).name;
    [song, Fs] = audioread(strcat(path, '/', filename));
    song = song(:,1) + song(:,2);
    song = resample(song, 20000, Fs);
    Fs = 20000;
    for j = 1:5:125
        test = song(Fs*j:Fs*(j+5), 1);
        spec_test = abs(spectrogram(test));
        spec_test = reshape(spec_test, 1, 8*32769);
        data = [data spec_test'];
    end
end

%% Plotting Dominant Spectrogram Modes
[u, s, v] = svd(data - mean(data(:)), 'econ');
plot(diag(s) ./ sum(diag(s)), 'ko');
plot3(v(1:50, 3), v(1:50, 4), v(1:50, 5), 'ko'), hold on
plot3(v(51:100, 3), v(51:100, 4), v(51:100, 5), 'ro')
plot3(v(101:150, 3), v(101:150, 4), v(101:150, 5), 'go')
xlabel('Mode 3')
ylabel('Mode 4')
zlabel('Mode 5')

%% k-Nearest Neighbors
result_knn = [];
for i = 1:1000
    q1 = randperm(50);
    q2 = randperm(50);
    q3 = randperm(50);
    xptv = v(1:50, 3:5);
    xbach = v(51:100, 3:5);
    xjc = v(101:150, 3:5);
    xtrain = [xptv(q1(1:30), :) xbach(q2(1:30), :) xjc(q3(1:30),:)];
    xtest = [xptv(q1(31:end), :) xbach(q2(31:end), :) xjc(q3(31:end),:)];
```

```
    ctrain = [ones(30,1) 2*ones(30,1) 3*ones(30,1)];
    ctest = [ones(20,1) 2*ones(20,1) 3*ones(20,1)];

    index = knnsearch(xtrain, xtest, 'A', 3);
    pre = [];
    true = 0;
    for j = 1:length(index)
        ind = index(j,:);
        x0 = [ctrain(ind(1)) ctrain(ind(2)) ctrain(ind(3))];
        x0 = mode(x0);
        pre = [pre x0];
        if pre == ctest(j,1)
            true = true + 1;
        end
    end
    acc = true/length(index);
    subplot(3,1,1)
    bar(pre);
    title('kNN')
end
result_knn = mean(acc);

%% Naive Bayes
result_nb = [];
for i = 1:1000
    q1 = randperm(50);
    q2 = randperm(50);
    q3 = randperm(50);
    xptv = v(1:50, 3:5);
    xbach = v(51:100, 3:5);
    xjc = v(101:150, 3:5);
    xtrain = [xptv(q1(1:30), :) xbach(q2(1:30), :) xjc(q3(1:30),:)];
    xtest = [xptv(q1(31:end), :) xbach(q2(31:end), :) xjc(q3(31:end),:)];
    ctrain = [ones(30,1) 2*ones(30,1) 3*ones(30,1)];
    ctest = [ones(20,1) 2*ones(20,1) 3*ones(20,1)];

    nb = fitcnb(xtrain,ctrain);
    pre = nb.predict(xtest);
    true = 0;
    trials = size(xtest,1);
    for j = 1:trials
        if pre(j,1) == ctest(j,1)
            true = true + 1;
        end
```

```
    end
    acc = true/trials;
    subplot(3,1,2)
    bar(pre)
    title('Naive Bayes')
end
result_nb = mean(acc);

%% Linear Discriminant Analysis
result_lda = [];
for i = 1:1000
    q1 = randperm(50);
    q2 = randperm(50);
    q3 = randperm(50);
    xptv = v(1:50, 3:5);
    xbach = v(51:100, 3:5);
    xjc = v(101:150, 3:5);
    xtrain = [xptv(q1(1:30), :) xbach(q2(1:30), :) xjc(q3(1:30),:)];
    xtest = [xptv(q1(31:end), :) xbach(q2(31:end), :) xjc(q3(31:end),:)];
    ctrain = [ones(30,1) 2*ones(30,1) 3*ones(30,1)];
    ctest = [ones(20,1) 2*ones(20,1) 3*ones(20,1)];

    pre = classify(xtest,xtrain,ctrain);
    true = 0;
    trials = size(xtest,1);
    for j = 1:trials
        if pre(j,1) == ctest(j,1)
            true = true + 1;
        end
    end
    acc = true/trials;
    subplot(3,1,3)
    bar(pre)
    title('LDA')
end
result_lda = mean(acc);
```