AMATH482 HOMEWORK4, MUSIC CLASSIFICATIONN

CIHAN MERTOZ

Abstract. In this report, we will analyse 9 different audio files using machine learning algorithms to do predictions. After analyzing three different cases, we will decide whether these algorithms are good enough to analyze the differences between different audio files

1. Introduction and Overview

There are three different algorithms that we will use in this analysis. First is the Linear Discriminant Analysis.Second one is the Supporting Vector Machine and lastly the Naive Bayes algorithm. All of these three methods have different benefits when compared to each other.

2. Theoretical Background

2.1. **Linear Discriminant Analysis.** Linear Discriminant analysis is a commonly used in supervised classification. It enables different labeled data to have clear separation points between their distribution points. The main goal is to find a good projection that maximizes the separation between the inter-class data.

2.2. **K-nearest neighbors.** K-nearest neighbors looks for the K nearest neighbors by searching for the nearest neighbors. It is also a supervised algorithm.

2.3. **Naive Bayes.** Naive Bayes is useful when analysing binary situations. It uses a probabilistic approach.

3. Algorithm

After downloading 18 different songs using YouTube and converting them to mp3 files we read the files using the audioread command. For each test we used 6 different songs. In the first test we used three very distinguishable songs whereas in the other two cases we used more similar songs. In second test, we used songs from the same area(Seattle) and in the last one we used songs from the same genre. After reading the files, we created a spectrogram for each song using the spectrogram command from Matlab. After creating 6 different matrices we concatenated the matrices together to perform singular value decomposition. After SVD, we used the diag to isolate the values to further use in our machine learning algorithms. Using the isolated values, we trained SVM, Naive Bayes and k-nearest neighbors using the built-in commands in Matlab.For each test we used the same exact code only changing the songs.

Date: March 2020.

4. Computational Results

4.1. **test1.** Here are the results for the test1. Since genres are different, our analysis clearly separates the 3 different sets.The accuracy of the algorithms is as follows: neighbors = 0.8167, naive = 0.8500, linear = 0.7833.

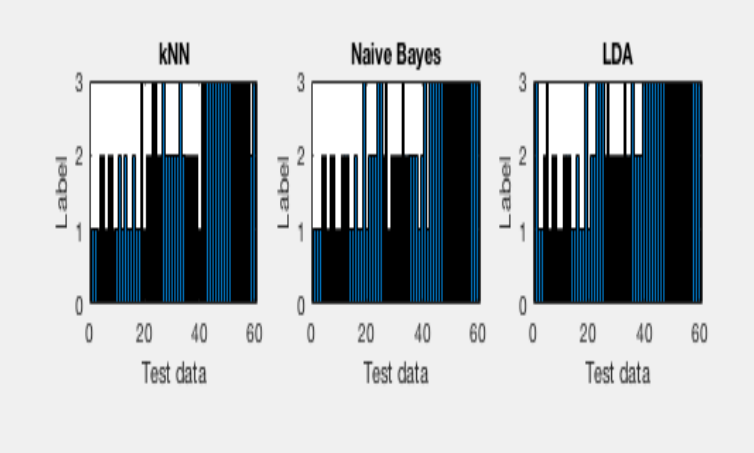

Figure 1. ML Analysis,test 1

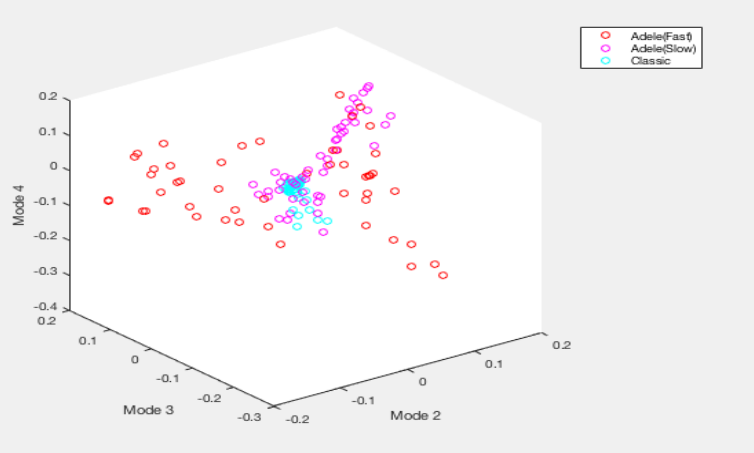

Figure 2. SDV Analysis,test 1

4.2. **test2.** Here are the results for the test2. Since we are dealing with similar songs there is more overlap compared to test 1. The accuracy of the algorithms is as follows: neighbors = 0.9500, naive = 0.7500, linear = 0.6167

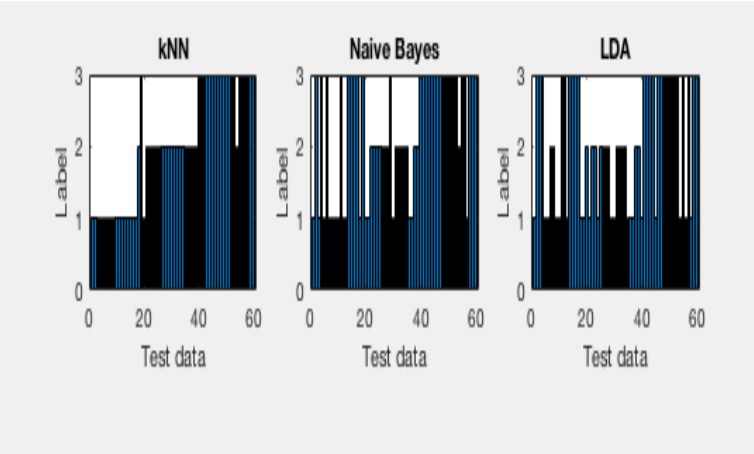

FIGURE 3. ML Analysis,test 2

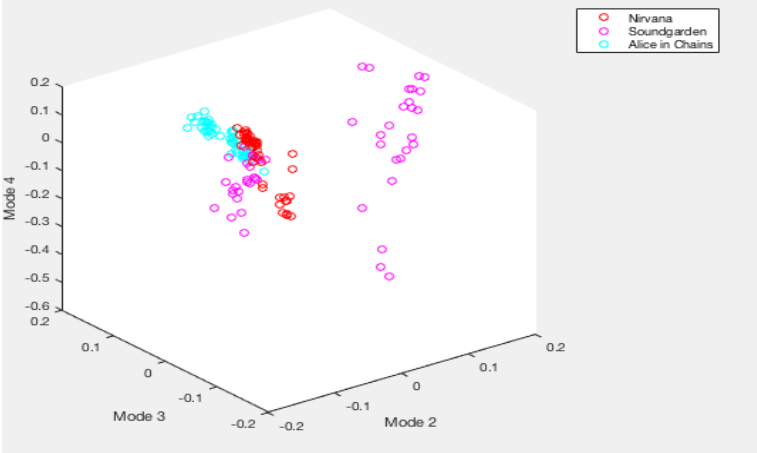

FIGURE 4. SDV Analysis,test 2

4.3. **test3.** Here are the results for the test3. Since we are dealing with the songs form the same genre there is a lot of overlap and the algorithm is doing more poorly to distinguish the songs. The accuracy of the algorithms is as follows: neighbors = 0.8333, naive = 0.7333, linear = 0.5500

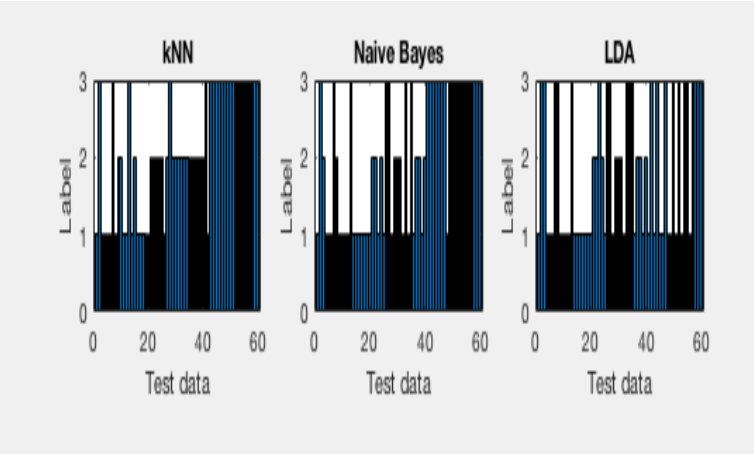

FIGURE 5. ML Analysis,test 3

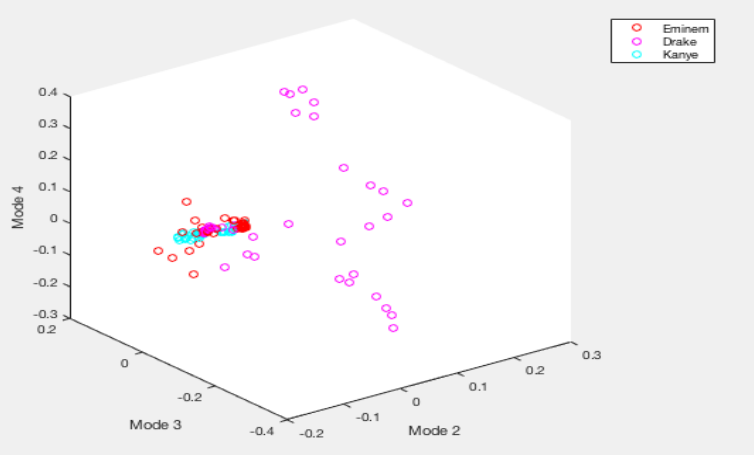

FIGURE 6. SDV Analysis,test 3

5. CONCLUSION

In this assignment we analyzed 18 different songs using common machine learning techniques. As we can see, some techniques are better than others depending on the situation. Therefore it is important to have a good overview of the situation to pick the right tools for the analysis.

6. Appendix A

[y,Fs] = audioread(filename) reads data from the file named filename, and returns sampled data, y, and a sample rate for that data, Fs. Idx = knnsearch(X,Y) finds the nearest neighbor in X for each query point in Y and returns the indices of the nearest neighbors in Idx, a column vector. Idx has the same number of rows as Y. [u,s,v] = svd() produces an economy-size decomposition of m by n matrix A.

7. Appendix B

```
% TEST 1

file = 'Adele1.mp3';
[song1,Fs1]=audioread(file);
song1 = song1'/ 2;
song1 = song1(1,:) + song1(2,:);
file = 'Adele2.mp3';
[song2,Fs2]=audioread(file);
song2 = song2'/ 2;
song2 = song2(1,:) + song2(2,:);
file = 'Adele3.mp3';
[song3,Fs3]=audioread(file);
song3 = song3'/ 2;
song3 = song3(1,:) + song3(2,:);


file = 'Adele4.mp3';
[song4,Fs4]=audioread(file);
song4 = song4'/ 2;
song4 = song4(1,:) + song4(2,:);

file = 'bach1.mp3';
[song5,Fs5]=audioread(file);
song5 = song5'/ 2;
song5 = song5(1,:) + song5(2,:);

file = 'bach2.mp3';
[song6,Fs6]=audioread(file);
song6 = song6'/ 2;
song6 = song6(1,:) + song6(2,:);

file = 'bach3.mp3';
[song7,Fs7]=audioread(file);
song7 = song7'/ 2;
song7 = song7(1,:) + song7(2,:);
```

```
A1 = [];
for kk = 40:5:160
    test = song1(1, Fs1*kk : Fs1*(kk+5));
%   test = resample(test, 20000, Fs1);
    vector = abs(spectrogram(test));
    vector = reshape(vector, [1, 8*32769]);
    A1 = [A1;vector];
end
A2 = [];
for kk = 40:5:160
    test = song2(1, Fs2*kk : Fs2*(kk+5));
%   test = resample(test, 20000, Fs1);
    vector = abs(spectrogram(test));
    vector = reshape(vector, [1, 8*32769]);
    A2 = [A2;vector];
end
A3 = [];
for kk = 40:5:160
    test = song3(1, Fs3*kk : Fs3*(kk+5));
%   test = resample(test, 20000, Fs1);
    vector = abs(spectrogram(test));
    vector = reshape(vector, [1, 8*32769]);
    A3 = [A3;vector];
end
A4 = [];
for kk = 40:5:160
    test = song4(1, Fs4*kk : Fs4*(kk+5));
%   test = resample(test, 20000, Fs1);
    vector = abs(spectrogram(test));
    vector = reshape(vector, [1, 8*32769]);
    A4 = [A4;vector];
end
A5 = [];
for kk = 40:5:160
    test = song5(1, Fs5*kk : Fs5*(kk+5));
%   test = resample(test, 20000, Fs1);
    vector = abs(spectrogram(test));
    vector = reshape(vector, [1, 8*32769]);
    A5 = [A5;vector];
end
A6 = [];
for kk = 40:5:160
    test = song7(1, Fs7*kk : Fs7*(kk+5));
%   test = resample(test, 20000, Fs1);
```

```
    vector = abs(spectrogram(test));
    vector = reshape(vector, [1, 8*32769]);
    A6 = [A6;vector];
end

A = cat(1,A1,A2,A3,A4,A5,A6);

A = A';


[u, s, v] = svd(A - mean(A(:)), 'econ');
plot(diag(s) ./ sum(diag(s)), 'ro');

thr = v';
plot3(thr(2, 1:50), thr(3, 1:50), thr(4, 1:50), 'ro'); hold on;
plot3(thr(2, 51:100), thr(3, 51:100), thr(4, 51:100), 'mo'); hold on;
plot3(thr(2, 101:150), thr(3, 101:150), thr(4, 101:150), 'co');
legend('Adele(Fast)', 'Adele(Slow)', 'Classic')
xlabel('Mode 2'); ylabel('Mode 3'); zlabel('Mode 4')


neighbors = []; naive = []; linear = [];
for test_trail = 1:1
    true = [ones(20,1); 2*ones(20,1); 3*ones(20,1)];
    q1 = randperm(50); q2 = randperm(50); q3 = randperm(50);
    x1 = v(1:50, 2:4);
    x2 = v(51:100, 2:4);
    x3 = v(101:150, 2:4);
    trainx = [x1(q1(1:30), :); x2(q2(1:30), :); x3(q3(1:30),:)];
    testx = [x1(q1(31:end), :); x2(q2(31:end), :); x3(q3(31:end),:)];
    % knn
    ind = knnsearch(trainx, testx);
    for i = 1:length(ind)
        if ind(i) <= 30
            ind(i) =  1;
        elseif ind(i) <= 60
            ind(i) = 2;
        else
            ind(i) = 3;
        end
    end
    temp = [ind==true];
    neighbors(test_trail) = sum(temp) / length(temp);
    subplot(3,3,1)
    bar(ind);
```

```
    title('kNN');
    xlabel('Test data'); ylabel('Label');

    % naive bayes
    ctrain = [ones(30,1); 2*ones(30,1); 3*ones(30,1)];
    nb = fitcnb(trainx, ctrain);
    pre = nb.predict(testx);
    temp = [pre== true];
    naive(test_trail) = sum(temp) / length(temp);
    subplot(3,3,2);
    bar(pre)
    title('Naive Bayes');
    xlabel('Test data'); ylabel('Label');

    % classify (Built in)
    pre = classify(testx, trainx, ctrain);
    temp = [pre== true];
    linear(test_trail) = sum(temp) / length(temp);
    subplot(3,3,3);
    bar(pre);
    title('LDA');
    xlabel('Test data'); ylabel('Label');
end

neighbors = mean(neighbors);
naive = mean(naive);
linear = mean(linear);

% TEST 2

file = 'Nirvana1.mp3';
[song1,Fs1]=audioread(file);
song1 = song1'/ 2;
song1 = song1(1,:) + song1(2,:);
file = 'Nirvana2.mp3';
[song2,Fs2]=audioread(file);
song2 = song2'/ 2;
song2 = song2(1,:) + song2(2,:);
file = 'SG1.mp3';
[song3,Fs3]=audioread(file);
song3 = song3'/ 2;
song3 = song3(1,:) + song3(2,:);


file = 'SG2.mp3';
```

```
[song4,Fs4]=audioread(file);
song4 = song4'/ 2;
song4 = song4(1,:) + song4(2,:);

file = 'Alice1.mp3';
[song5,Fs5]=audioread(file);
song5 = song5'/ 2;
song5 = song5(1,:) + song5(2,:);

file = 'Alice2.mp3';
[song7,Fs7]=audioread(file);
song7 = song7'/ 2;
song7 = song7(1,:) + song7(2,:);


A1 = [];
for kk = 40:5:160
    test = song1(1, Fs1*kk : Fs1*(kk+5));
%   test = resample(test, 20000, Fs1);
    vector = abs(spectrogram(test));
    vector = reshape(vector, [1, 8*32769]);
    A1 = [A1;vector];
end
A2 = [];
for kk = 40:5:160
    test = song2(1, Fs2*kk : Fs2*(kk+5));
%   test = resample(test, 20000, Fs1);
    vector = abs(spectrogram(test));
    vector = reshape(vector, [1, 8*32769]);
    A2 = [A2;vector];
end
A3 = [];
for kk = 40:5:160
    test = song3(1, Fs3*kk : Fs3*(kk+5));
%   test = resample(test, 20000, Fs1);
    vector = abs(spectrogram(test));
    vector = reshape(vector, [1, 8*32769]);
    A3 = [A3;vector];
end
A4 = [];
for kk = 40:5:160
    test = song4(1, Fs4*kk : Fs4*(kk+5));
%   test = resample(test, 20000, Fs1);
    vector = abs(spectrogram(test));
    vector = reshape(vector, [1, 8*32769]);
```

```
    A4 = [A4;vector];
end
A5 = [];
for kk = 40:5:160
    test = song5(1, Fs5*kk : Fs5*(kk+5));
%   test = resample(test, 20000, Fs1);
    vector = abs(spectrogram(test));
    vector = reshape(vector, [1, 8*32769]);
    A5 = [A5;vector];
end
A6 = [];
for kk = 40:5:160
    test = song7(1, Fs7*kk : Fs7*(kk+5));
%   test = resample(test, 20000, Fs1);
    vector = abs(spectrogram(test));
    vector = reshape(vector, [1, 8*32769]);
    A6 = [A6;vector];
end

A = cat(1,A1,A2,A3,A4,A5,A6);

A = A';


[u, s, v] = svd(A - mean(A(:)), 'econ');
plot(diag(s) ./ sum(diag(s)), 'ro');

thr = v';
plot3(thr(2, 1:50), thr(3, 1:50), thr(4, 1:50), 'ro'); hold on;
plot3(thr(2, 51:100), thr(3, 51:100), thr(4, 51:100), 'mo'); hold on;
plot3(thr(2, 101:150), thr(3, 101:150), thr(4, 101:150), 'co');
legend('Nirvana', 'Soundgarden', 'Alice in Chains')
xlabel('Mode 2'); ylabel('Mode 3'); zlabel('Mode 4')


neighbors = []; naive = []; linear = [];
for test_trail = 1:1
    true = [ones(20,1); 2*ones(20,1); 3*ones(20,1)];
    q1 = randperm(50); q2 = randperm(50); q3 = randperm(50);
    x1 = v(1:50, 2:4);
    x2 = v(51:100, 2:4);
    x3 = v(101:150, 2:4);
    trainx = [x1(q1(1:30), :); x2(q2(1:30), :); x3(q3(1:30),:)];
    testx = [x1(q1(31:end), :); x2(q2(31:end), :); x3(q3(31:end),:)];
    % knn
```

```
    ind = knnsearch(trainx, testx);
    for i = 1:length(ind)
        if ind(i) <= 30
            ind(i) = 1;
        elseif ind(i) <= 60
            ind(i) = 2;
        else
            ind(i) = 3;
        end
    end
    temp = [ind==true];
    neighbors(test_trail) = sum(temp) / length(temp);
    subplot(3,3,1)
    bar(ind);
    title('kNN');
    xlabel('Test data'); ylabel('Label');

    % naive bayes
    ctrain = [ones(30,1); 2*ones(30,1); 3*ones(30,1)];
    nb = fitcnb(trainx, ctrain);
    pre = nb.predict(testx);
    temp = [pre== true];
    naive(test_trail) = sum(temp) / length(temp);
    subplot(3,3,2);
    bar(pre)
    title('Naive Bayes');
    xlabel('Test data'); ylabel('Label');

    % classify (Built in)
    pre = classify(testx, trainx, ctrain);
    temp = [pre== true];
    linear(test_trail) = sum(temp) / length(temp);
    subplot(3,3,3);
    bar(pre);
    title('LDA');
    xlabel('Test data'); ylabel('Label');
end

neighbors = mean(neighbors);
naive = mean(naive);
linear = mean(linear);

% TEST 3

file = 'Drake1.mp3';
```

```
[song1,Fs1]=audioread(file);
song1 = song1'/ 2;
song1 = song1(1,:) + song1(2,:);
file = 'Drake2.mp3';
[song2,Fs2]=audioread(file);
song2 = song2'/ 2;
song2 = song2(1,:) + song2(2,:);
file = 'Eminem1.mp3';
[song3,Fs3]=audioread(file);
song3 = song3'/ 2;
song3 = song3(1,:) + song3(2,:);


file = 'Eminem2.mp3';
[song4,Fs4]=audioread(file);
song4 = song4'/ 2;
song4 = song4(1,:) + song4(2,:);

file = 'Kanye3.mp3';
[song5,Fs5]=audioread(file);
song5 = song5'/ 2;
song5 = song5(1,:) + song5(2,:);

file = 'Kanye2.mp3';
[song7,Fs7]=audioread(file);
song7 = song7'/ 2;
song7 = song7(1,:) + song7(2,:);


A1 = [];
for kk = 40:5:160
    test = song1(1, Fs1*kk : Fs1*(kk+5));
    vector = abs(spectrogram(test));
    vector = reshape(vector, [1, 8*32769]);
    A1 = [A1;vector];
end
A2 = [];
for kk = 40:5:160
    test = song2(1, Fs2*kk : Fs2*(kk+5));
    vector = abs(spectrogram(test));
    vector = reshape(vector, [1, 8*32769]);
    A2 = [A2;vector];
end
A3 = [];
for kk = 40:5:160
```

```
    test = song3(1, Fs3*kk : Fs3*(kk+5));
    vector = abs(spectrogram(test));
    vector = reshape(vector, [1, 8*32769]);
    A3 = [A3;vector];
end
A4 = [];
for kk = 40:5:160
    test = song4(1, Fs4*kk : Fs4*(kk+5));

    vector = abs(spectrogram(test));
    vector = reshape(vector, [1, 8*32769]);
    A4 = [A4;vector];
end
A5 = [];
for kk = 40:5:160
    test = song5(1, Fs5*kk : Fs5*(kk+5));
    vector = abs(spectrogram(test));
    vector = reshape(vector, [1, 8*32769]);
    A5 = [A5;vector];
end
A6 = [];
for kk = 40:5:160
    test = song7(1, Fs7*kk : Fs7*(kk+5));
    vector = abs(spectrogram(test));
    vector = reshape(vector, [1, 8*32769]);
    A6 = [A6;vector];
end

A = cat(1,A1,A2,A3,A4,A5,A6);

A = A';


[u, s, v] = svd(A - mean(A(:)), 'econ');
plot(diag(s) ./ sum(diag(s)), 'ro');

thr = v';
plot3(thr(2, 1:50), thr(3, 1:50), thr(4, 1:50), 'ro'); hold on;
plot3(thr(2, 51:100), thr(3, 51:100), thr(4, 51:100), 'mo'); hold on;
plot3(thr(2, 101:150), thr(3, 101:150), thr(4, 101:150), 'co');
legend('Eminem', 'Drake', 'Kanye')
xlabel('Mode 2'); ylabel('Mode 3'); zlabel('Mode 4')


neighbors = []; naive = []; linear = [];
```

```
for test_trail = 1:1
    true = [ones(20,1); 2*ones(20,1); 3*ones(20,1)];
    q1 = randperm(50); q2 = randperm(50); q3 = randperm(50);
    x1 = v(1:50, 2:4);
    x2 = v(51:100, 2:4);
    x3 = v(101:150, 2:4);
    trainx = [x1(q1(1:30), :); x2(q2(1:30), :); x3(q3(1:30),:)];
    testx = [x1(q1(31:end), :); x2(q2(31:end), :); x3(q3(31:end),:)];

    ind = knnsearch(trainx, testx);
    for i = 1:length(ind)
       if ind(i) <= 30
           ind(i) =  1;
       elseif ind(i) <= 60
           ind(i) = 2;
       else
           ind(i) = 3;
       end
    end
    temp = [ind==true];
    neighbors(test_trail) = sum(temp) / length(temp);
    subplot(3,3,1)
    bar(ind);
    title('kNN');
    xlabel('Test data'); ylabel('Label');


    ctrain = [ones(30,1); 2*ones(30,1); 3*ones(30,1)];
    nb = fitcnb(trainx, ctrain);
    pre = nb.predict(testx);
    temp = [pre== true];
    naive(test_trail) = sum(temp) / length(temp);
    subplot(3,3,2);
    bar(pre)
    title('Naive Bayes');
    xlabel('Test data'); ylabel('Label');


    pre = classify(testx, trainx, ctrain);
    temp = [pre== true];
    linear(test_trail) = sum(temp) / length(temp);
    subplot(3,3,3);
    bar(pre);
    title('LDA');
    xlabel('Test data'); ylabel('Label');
```

```
end

neighbors = mean(neighbors);
naive = mean(naive);
linear = mean(linear);
```