Contents

```
clear all; close all; clc;
% Same operations like the test 1 & test 2, but we change the class to same genres different artist.

% Load music data, sort and reduce sampleling rate.
% Section out 5 seconds music data from each Genre's song.
% Genre :"Hip-Hop", "Classical","Rock" ;
% 5 songs for each Genre.

tic
% Read folder and file name.
flist = dir("music");
fname = {flist.name};
fname = fname(~strncmp(fname, '.', 1));

% Data aquare, format and reduce for each song.
musicAll = [];
lables = [];
Aname = ["Hip-Hop", "Classical","Rock"];

for h = 1:length(Aname)
    localAname = Aname(h);
   % using artist name to sort song
    tf = startsWith(fname,localAname,'IgnoreCase',true);
    target = fname(tf);
    for i = 1:length(target)
        % load song
        localsname = string(target(i));
        [music,Fs] = audioread(strcat ("music\",localsname));

        %lower sampling rate 1/10
        Fs = Fs/10;
        music = music(1:10:end,:);

        %convert to mono tone; skip if already mono tone.
        musicSize = size(music);
        musicSize = musicSize(2);
        if musicSize == 1
            Mmono = music;
        else
            Mmono = zeros(length(music),1);
            for j = 1:length(music)
                if(music(j,1) == 0 || music(j,2) == 0)
                    Mmono(j,1) = max(music(j,:));
                else
                    Mmono(j,1) = (music(j,1)+music(j,2))/2;
                end
            end
        end

        % remove zeros on head and tail
        Mhead = find(Mmono,1,"first");
        Mtail = find(Mmono,1,"last");
        Mmono = Mmono(Mhead:Mtail);

        % aquare 5 seconds clips from music; every 5sec one clip
        % round music to n * 5 seconds
        Intervals = Fs*5;
        sections = floor(length(Mmono)/(Fs*5));

        Mmono = Mmono(1:(Intervals * sections),1);

        data5s = reshape(Mmono, [Intervals,sections]);

        musicAll = [musicAll,data5s];
        lables = [lables; repmat(localAname,size(data5s,2),1)];
    end
end
lables = lables.';
trainlabel = lables;

toc
```

```
历时 10.980400 秒。
```

start rearrange data and fit

```
tic
% svd data
[u,s,v] = svd(abs(fft(musicAll)),"econ");
sig = diag(s);
lambda = sig.^2;
%plot ( cumsum ( lambda /sum ( lambda )), 'bo ')
utrunc = u(:, 1:196);
traindata = utrunc'*musicAll;

%building test data set by selection rate of 1/10 from traindata set,
%randomly selected.

sizen = floor(size(traindata,2)/(10));
testnumber = randperm (size(traindata,2), sizen);

testdata = traindata(:,testnumber);
testlabel = trainlabel(:,testnumber);

%remove test data from train data set.
traindata(:,testnumber) = [];
trainlabel(:,testnumber) = [];

%non svd train and test data sets.
nonSvdtraindata = abs(fft(musicAll));
nonSvdtestdata = nonSvdtraindata(:,testnumber);
%remove test data from train data set.
nonSvdtraindata(:,testnumber) = [];

toc
```

```
历时 2.933886 秒。
```

modeling in svd data and non svd data

```
tic
% Naive Bayes classifier modeling(svd data)
NBCModel = fitcnb(traindata.', trainlabel);
NBCLoss = loss(NBCModel , testdata.', testlabel);

svdDataTimeElapsed = toc;

tic
% Naive Bayes classifier modeling(non svd data)
nonSvdNBCModel = fitcnb(nonSvdtraindata.', trainlabel);
nonSvdNBCLoss = loss(nonSvdNBCModel , nonSvdtestdata.', testlabel);

nonSvdDataTimeElapsed = toc;
```

```
%LDA classifier modeling(svd data)
tic
LDAtrainData = traindata.';
LDAtrainlabel = trainlabel.';
MdlLinear = fitcdiscr(LDAtrainData,LDAtrainlabel);
Testclass = predict(MdlLinear,testdata.');
ars = find(Testclass == testlabel.');
LDA_SVDrate = length(ars)/length(testlabel);

svdLDATimeElapsed = toc;

%LDA classifier modeling(non svd data)
tic
LDAtrainData = nonSvdtraindata.';
LDAtrainlabel = trainlabel.';
MdlLinear = fitcdiscr(LDAtrainData,LDAtrainlabel);
Testclass = predict(MdlLinear,nonSvdtestdata.');
ars = find(Testclass == testlabel.');
LDA_nonSVDrate = length(ars)/length(testlabel);

nonSvdLDATimeElapsed = toc;
```

plot

```
figure(1)
subplot(1,2,1)
b = bar(1-[NBCLoss,nonSvdNBCLoss]);
xtips1 = b(1).XEndPoints;
ytips1 = b(1).YEndPoints;
labels1 = string(b(1).YData);
ylim ([0, 1]) ;
text(xtips1,ytips1,labels1,'HorizontalAlignment','center',...
    'VerticalAlignment','bottom')
xticklabels (["SVD-Data","Non SVD-data"]);
ylabel (" Classification Accuracy ( Scale of 0 to 1) ");
xlabel (" Data Type ");
title (" Accuracy of Naive Bayes classifier Model of Three Different Genres");
grid on
subplot(1,2,2)
b = bar([svdDataTimeElapsed;nonSvdDataTimeElapsed]);
xtips1 = b(1).XEndPoints;
ytips1 = b(1).YEndPoints;
labels1 = string(b(1).YData);
text(xtips1,ytips1,labels1,'HorizontalAlignment','center',...
    'VerticalAlignment','bottom')
xticklabels (["SVD-Data time","Non SVD-data time"]);
ylabel (" Time(seconds) ");
xlabel (" Data Type ");
title (" Processing Time of Naive Bayes classifier Model of Three Different Genres");

figure(2)
subplot(1,2,1)
b = bar([LDA_SVDrate;LDA_nonSVDrate]);
xtips1 = b(1).XEndPoints;
ytips1 = b(1).YEndPoints;
labels1 = string(b(1).YData);
ylim ([0, 1]) ;
text(xtips1,ytips1,labels1,'HorizontalAlignment','center',...
    'VerticalAlignment','bottom')
xticklabels (["SVD-Data","Non SVD-data"]);
ylabel (" Classification Accuracy ( Scale of 0 to 1) ");
xlabel (" Data Type ");
title (" Accuracy of Linear discriminant analysis Model of Three Different Genres");
grid on
subplot(1,2,2)
b = bar([svdLDATimeElapsed;nonSvdLDATimeElapsed]);
xtips1 = b(1).XEndPoints;
ytips1 = b(1).YEndPoints;
labels1 = string(b(1).YData);
text(xtips1,ytips1,labels1,'HorizontalAlignment','center',...
    'VerticalAlignment','bottom')
xticklabels (["SVD-Data time","Non SVD-data time"]);
ylabel (" Time(seconds) ");
xlabel (" Data Type ");
title (" Processing Time of  Linear discriminant analysis classifier Model of Three Different Genres");
```