Homework 4: Music Classification

Yicheng Duan

AMATH 482

03/06/2020

Abstract

In this homework, we need develop a code that build models of different type of music, and let machine classify a given piece of music. First, we square training data from music and make a test data set. We then train machine using this training and find accuracy of our machine. We perform a singular value decomposition on our training data to reduce the redundancy of our training data, however we see this action may hugely impact the accuracy of our machine classification.

Introduction and Overview

We have three different test goal of our code. We first need classify our test sample with three different bands/artists in their own different genre. We then test out machine to identify our sample clips from three bands from same genre. Finally, we will train our machine to classify samples form three genres with multiple bands/artists.

In order to build our training data set, we slice the music we select from each type into 5 seconds clips. Then we label those clips with their type names, like "Dr. Dre", "Rock" or "Hip-Hop", in another array in order. Now, we built perfect data sets for a supervised machine learning, which is a good to classify and predict new data's category.

We need to construct a low-dimension featured data of our training data set, then put this low-dimension data set to our machine learning algorithms. We use PCA modes generated by SVD to determine the redundancy. We set a 95% level of energy of our training data as a qualified representative.

We use two classifiers in the classification and compare the results. We use Naive Bayes classifier and Linear discriminant analysis model to perform our classification. Each classifier was feed with both SVD reduced data and non SVD reduced data for comparison. Also, we count the processing time of each operation for comparison.

Theoretical Background

Principal component analysis (PCA) is a statistical procedure that uses an orthogonal transformation to convert a set of observations of possibly correlated variables into a set of values of linearly uncorrelated variables called principal components. This transformation is defined in such a way that the first principal component has the largest possible variance (that is, accounts for as much of the variability in the data as possible), and each succeeding component in turn has the highest variance possible under the constraint that it is orthogonal to the preceding components. (wiki)

The PCA allows for large, complex, and even somewhat random sets of data to be reduced to lower dimensions of dynamics without any knowledge of underlying behavior.

SVD, The Singular Value Decomposition, is an operation finds the diagonal values of variance in order form largest to smallest.

$$A = U\Sigma V^*$$

Here, U and V are unitary matrix and Σ is diagonal matrix.

In machine learning, **naïve Bayes classifiers** are a family of simple "probabilistic classifiers" based on applying Bayes' theorem with strong (naïve) independence assumptions between the features. They are among the simplest Bayesian network models. Naïve Bayes classifiers are highly scalable, requiring a number of parameters linear in the number of variables (features/predictors) in a learning problem. Maximum-likelihood training can be done by evaluating a closed-form expression,[5]:718 which takes linear time, rather than by expensive iterative approximation as used for many other types of classifiers. (wiki)

Using Bayes' theorem, the conditional probability can be decomposed as

$$p(C_k|X) = \frac{p(C_k)p(x|C_k)}{p(X)} \quad \text{(e.q.1)}$$
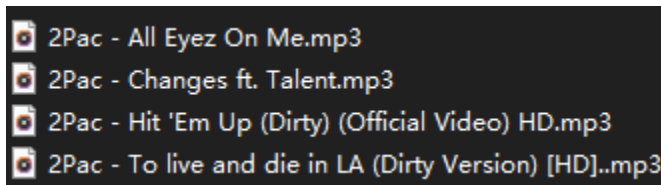
Which interprets as:

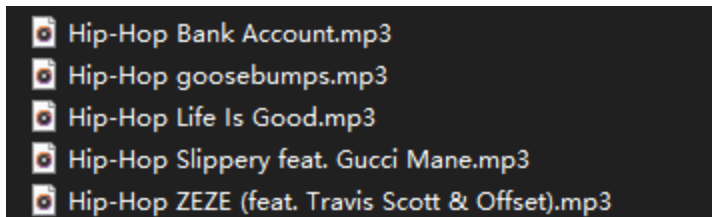$$posterior = \frac{prior \times likehood}{evidence} \quad \text{(e.q.2)}$$

Linear discriminant analysis (LDA), normal discriminant analysis (NDA), or discriminant function analysis is a generalization of Fisher's linear discriminant, a method used in statistics, pattern recognition, and machine learning to find a linear combination of features that characterizes or separates two or more classes of objects or events. The resulting combination may be used as a linear classifier, or, more commonly, for dimensionality reduction before later classification. (wiki)

Algorithm Development

We first import our audio file to MATLAB by using function *[music, Fs] = audioread(file)*. However, in order to find the file name by name or genre, we read the folder of music and using a loop to find music and then build a training data set. We order our music file as:

Which are [artists/bands] or [genre]+ name.mp3

In the main loop, we use function *startwith* to select the name we want from a list of file names listed by *dri(folder).name*. Then we load the song and lower it's sampling rate by 1/10. We just want the data to analyze so we mono toned the music if it is stereo and cut the zeros in the head and tail. Then we slice this music into 5 seconds clip by counting samples. We put all those clips into our main data set and label it with the band name or genre in another data set only for label. So, we can assess our label easy and efficiencies for later compare. (see code)

Then we perform a PCA to our data by SVD and store it to a train data set. We also remain our raw data sets for comparison. We then build a test data sets and label at 1/10 size of our train data size. We random select indexes and their data from the training data and remove those indexes and their data from our training data set. That prevent the cheating. Same action to non SVD training data sets.

We now train models by functions: *fitcnb* for naïve Bayes train and *loss* for find error of our naïve Bayes model by feed in our test data. Using *tic* and *toc* to measure processing time. Both SVD data and non SVD data are trained for comparison.

We use function *fitcdiscr* to train a Linear discriminant analysis model and *predict* to get the classification result of our model by feed in our test data. Then, we compare the result with the test data's original label to find the success rate.

We plot our result and compare.

Computational Results

Test 1:

We choose "Dr. Dre", "Beethoven" and "The Beatles" as our music source represent three different stars of different genres. We fist find a 95% critical level of data at Principal component mode at 226. (figure 1.1)
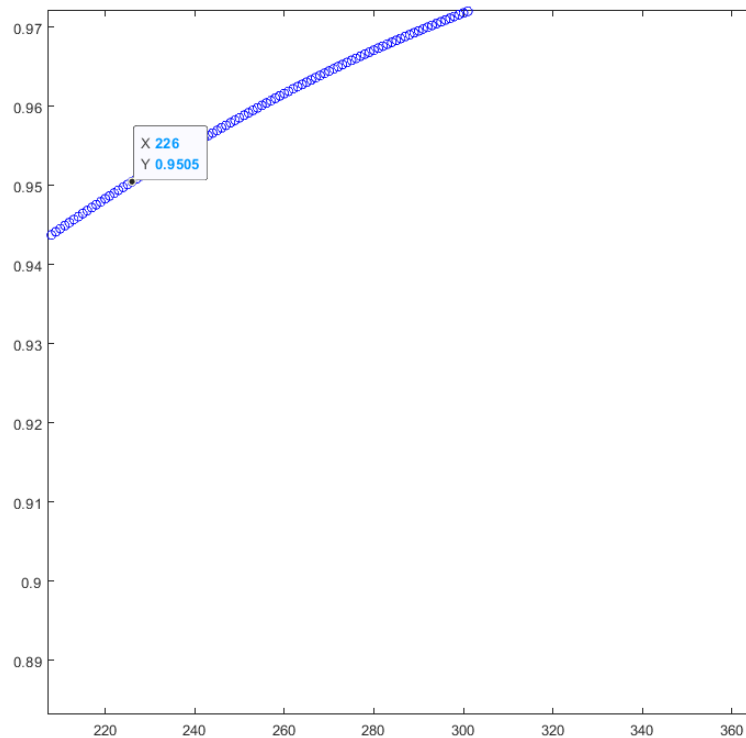
Figure -1.1

So we see a PCA mode 226 training data, trained by naïve Bayes classifier has a success classify rate of 0.75. The non PCA/SVD data has a successful classify rate of 0.90. In Linear discriminant analysis model, we have a success classify rate of 0.55 by SVD data training and 0.855 by non SVD data training. So, we see that SVD data saved a lot of processing time but got less accuracy in both naïve Bayes and LDA classifier. Naïve Bayes training perform better globally in this test. (figure 1.2)
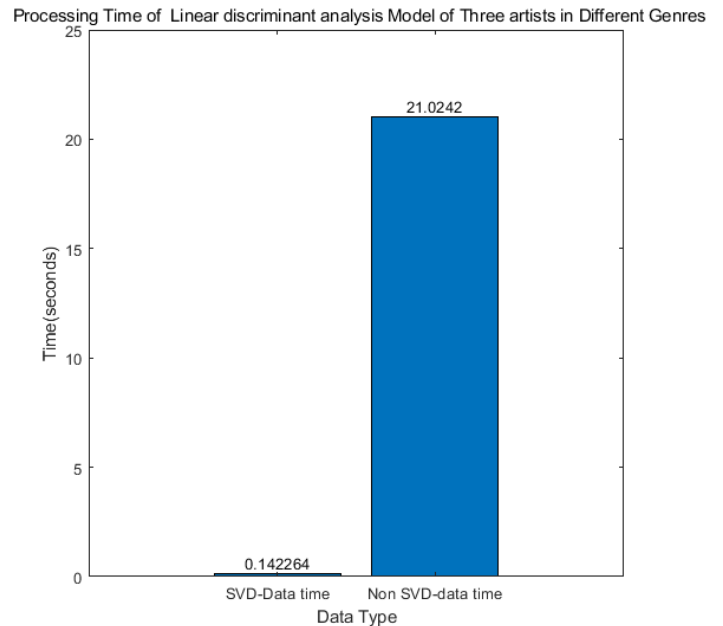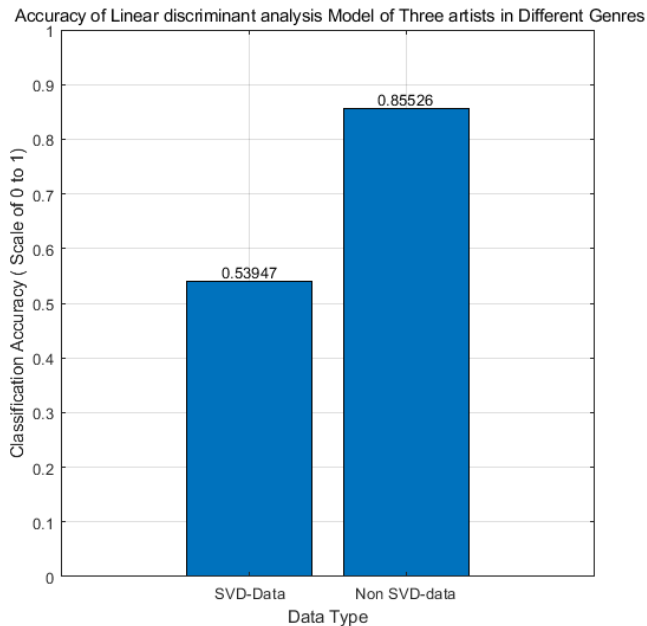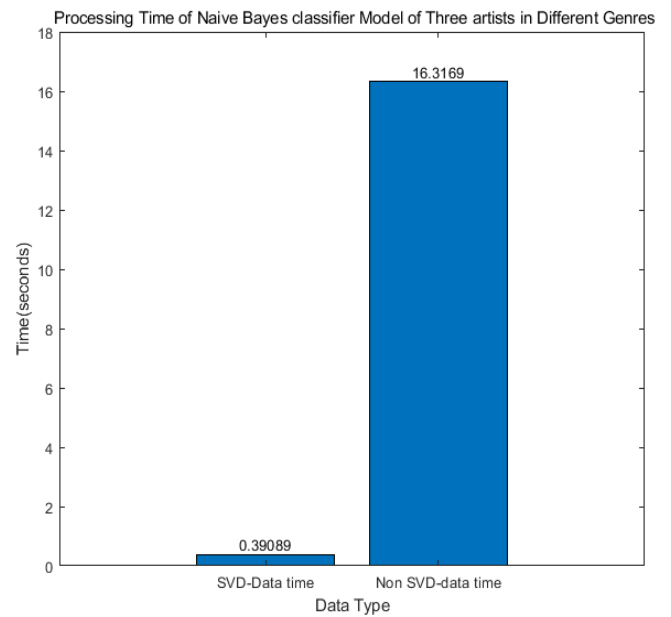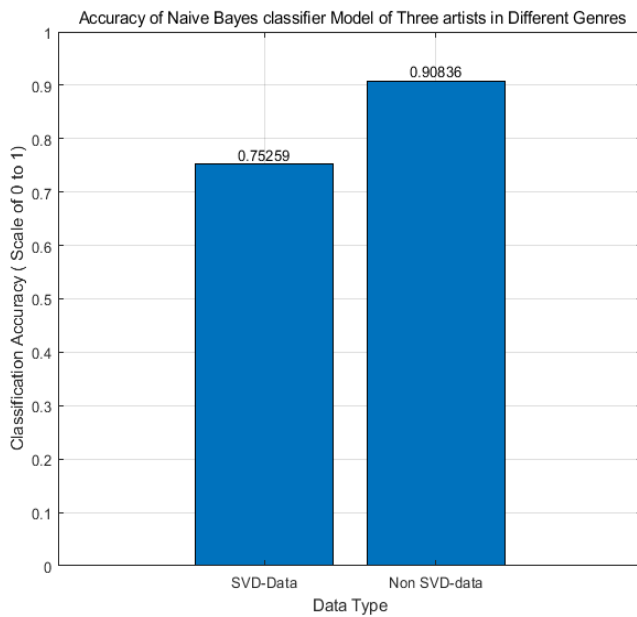

Figure 1.2

Test 2:

We choose "Dr. Dre", "Snoop Dogg" and "Tupac" as our music source represent three stars in same genres. We fist find a 95% critical level of data at Principal component mode at 226. (figure 2.1)
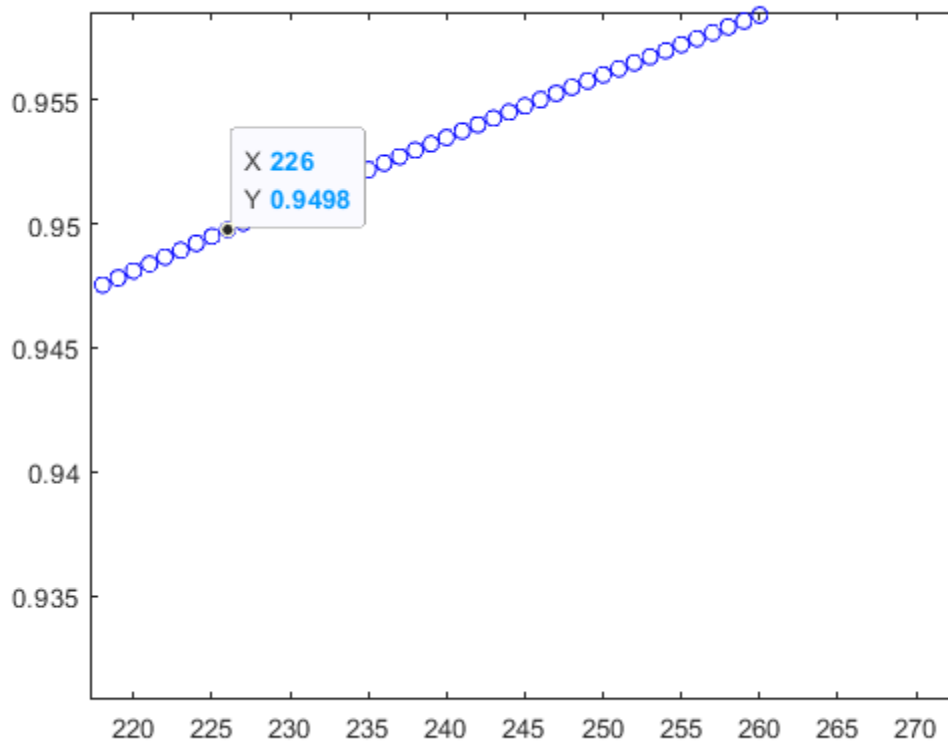

Figure 2.1

The model trained by naïve Bayes model have an accuracy of 0.43 in SVD data training and 0.57 in non SVD data training. The model trained by Linear discriminant analysis model have an accuracy of 0.22 in SVD data training and 0.76 in non SVD data training. The naïve Bayes performed bad in this test. In LDA modeling, accuracy of SVD data trained model is awful. Only non SVD data trained LDA model has a fair accuracy. This inaccuracy of machine learning may cause by the artist "Snoop Dogg". Although Snoop Dogg is a Hip-Hop star in 90s just like other two, his music contains a lot of non-musical sound like conversations. He also has a lot of variations of Hip-Hop elements inside his music. That may cause inconsistency inside our training data set. This inconsistency made our SVD data trained model even worse, since SVD and PCA are sensitive to variations. (figure 2.2)
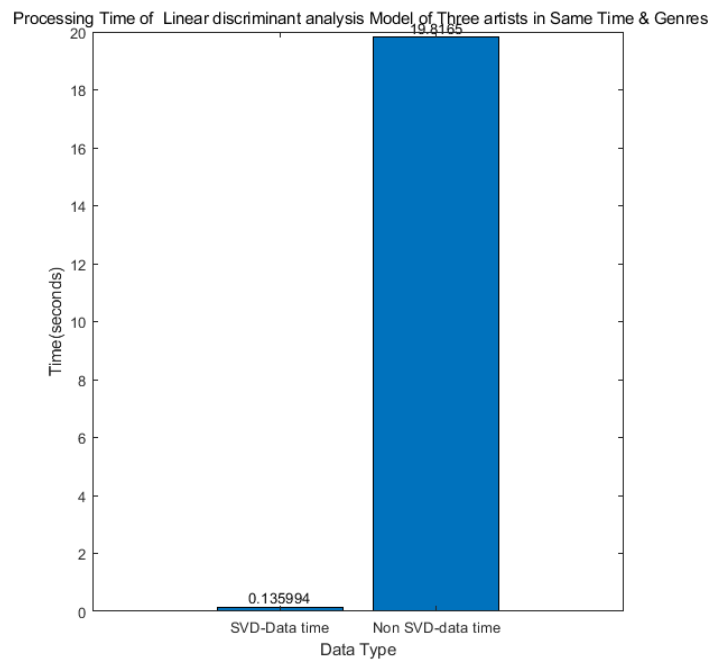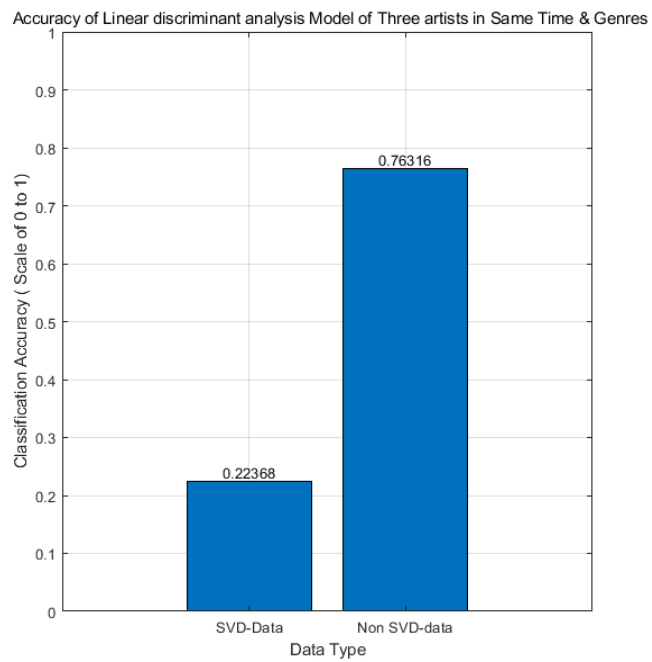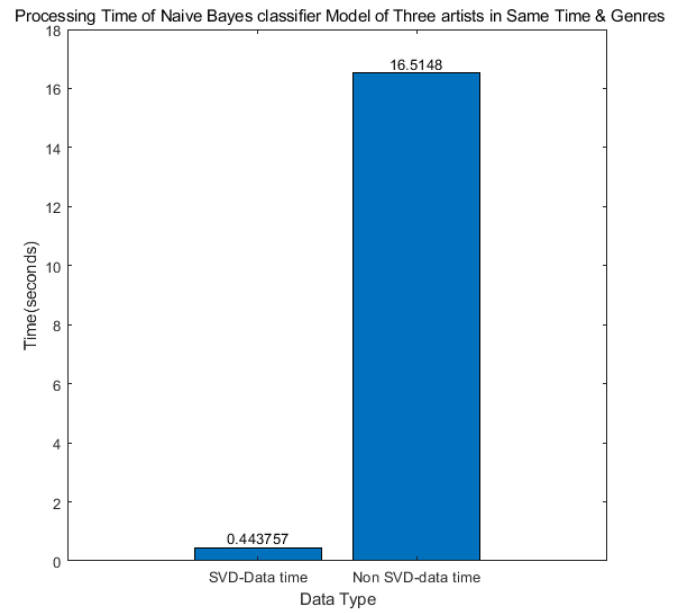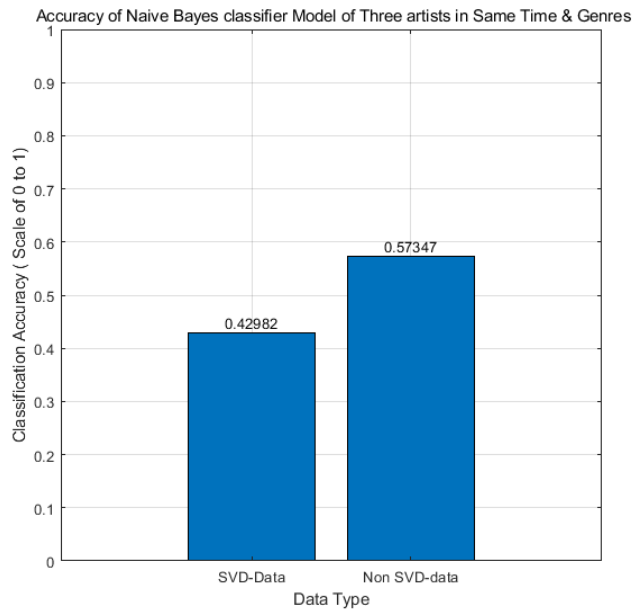
Figure 2.2

Test 3.

We choose "Hip-Hop", "Classical" and "Rock" as three genres. We choose 5 trap hip-hop music from different artists, 5 classical piano music from different artists, and 5 Rock music from different artists. No repetition occurred. We fist find a 95% critical level of data at Principal component mode at 196. (figure 3.1)
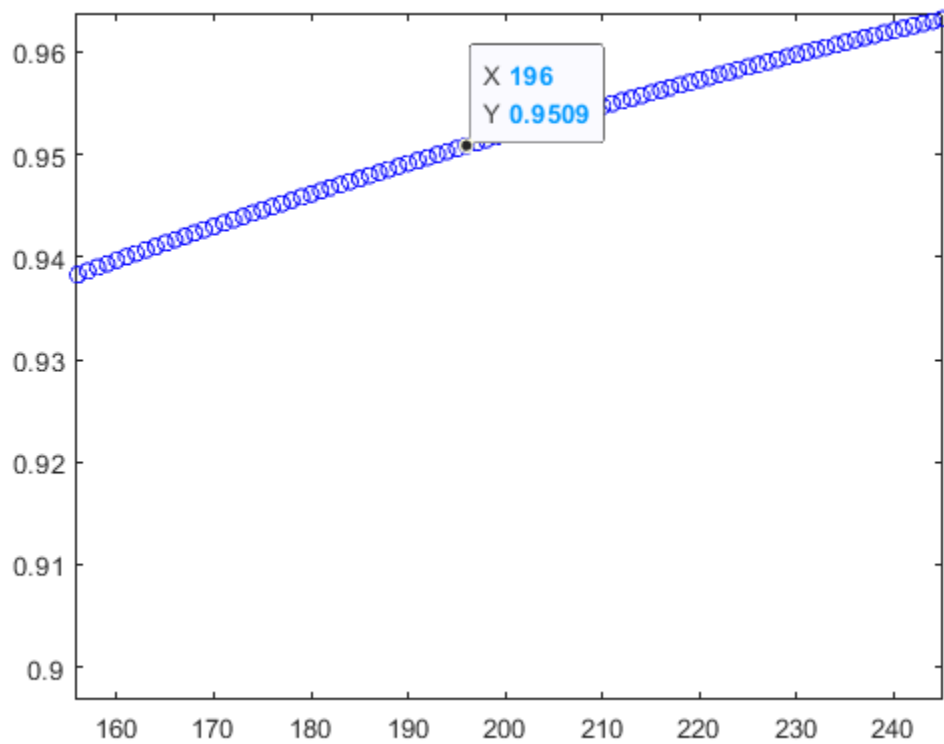

Figure 3.1

The model trained by naïve Bayes model have an accuracy of 0.88 in SVD data training and 0.88 in non SVD data training. The model trained by Linear discriminant analysis model have an accuracy of 0.49 in SVD data training and 0.98 in non SVD data training.

In this test, Both SCD data trained and non SVD data trained naïve Bayes model have a good accuracy. The model trained by Linear discriminant analysis has a bad score on SVD data training and an extremely good result on non SVD data training. (figure 3.2)
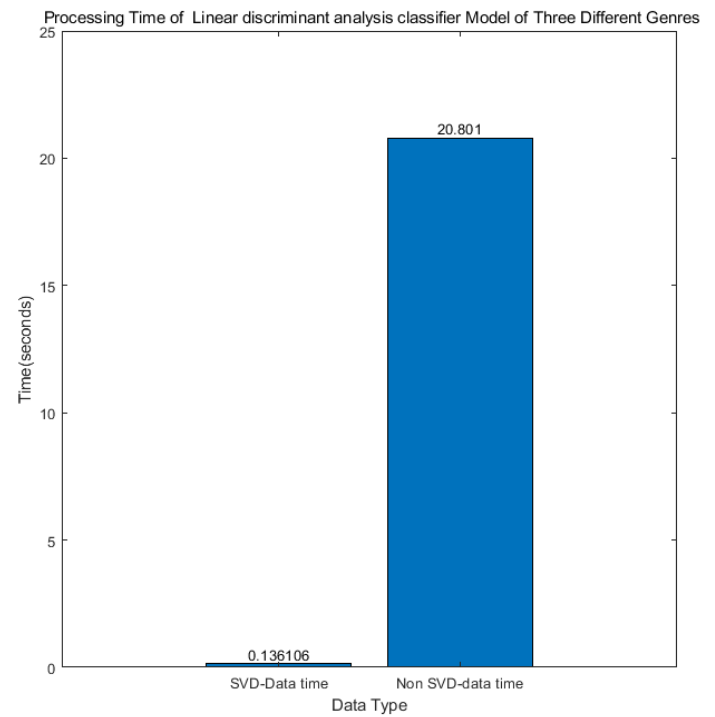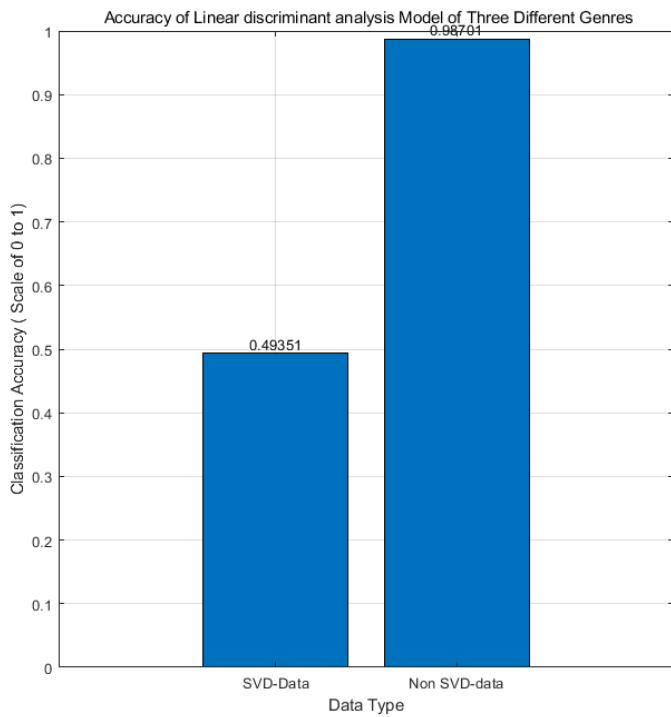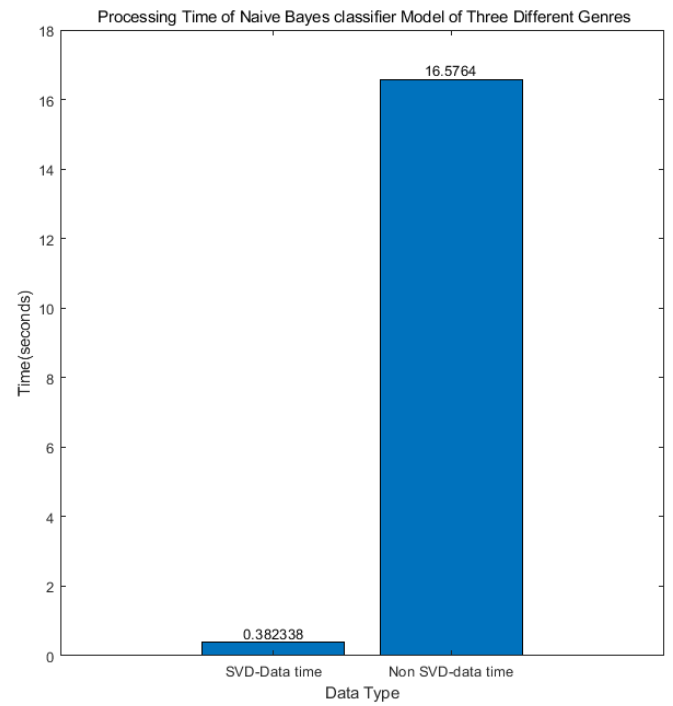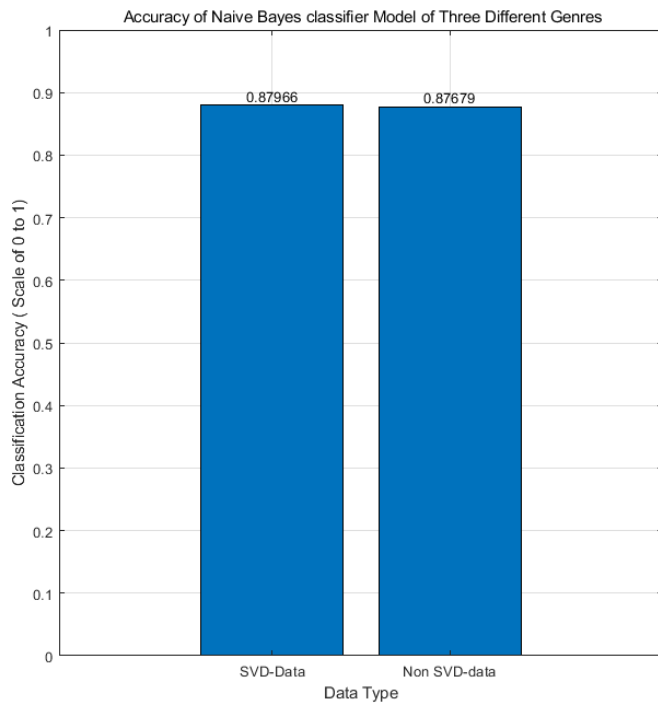
Figure 3.2

Summary and Conclusions

We can see that naïve Bayes perform better when you train model with SVD data. Linear discriminant analysis performs better when you train model with non SVD data. Indeed, PCA data by SVD with save you tons of proccing time. However, choosing proper training data and algorithm will earn machine learning more efficiency.

Index table of functions

`dir`	lists files and folders in the current folder
`tf = strncmp(s1,s2,n)`	strncmp(s1,s2,n) compares up to n characters of s1 and s2. The function returns 1 (true) if the two are identical and 0 (false) otherwise. Text is considered identical if the content of each is the same up to the end or the first n characters, whichever comes first. The return result tf is of data type logical. The first two input arguments can be any combination of string arrays, character vectors, and cell arrays of character vectors.
`startsWith(str,pattern)`	returns 1 (true) if str starts with the specified pattern and returns 0 (false) otherwise. If pattern is an array containing multiple patterns, then startsWith returns 1 if it finds that str starts with any element of pattern.
`[y,Fs] = audioread(filename)`	reads data from the file named filename, and returns sampled data, y, and a sample rate for that data, Fs.
`[u,s,v] = svd()`	performs a singular value decomposition of matrix A, such that A = U*S*V'.
`floor`	rounds each element of X to the nearest integer less than or equal to that element.
`randperm`	returns a row vector containing k unique integers selected randomly from 1 to n.
`fitcnb`	Mdl = fitcnb(Tbl,Y) returns a multiclass naive Bayes model (Mdl), trained by the predictors in the table Tbl and class labels in the array Y.
`loss`	L = loss(mdl,X,Y) returns a scalar representing how well mdl classifies the data in X when Y contains the true classifications. When computing the loss, the loss function normalizes the class probabilities in Y to the class probabilities used for training, which are stored in the Prior property of mdl.

`fitcdiscr`	Mdl = fitcdiscr(Tbl,Y) returns a fitted discriminant analysis model based on the input variables contained in the table Tbl and response Y.
`predict`	topicIdx = predict(ldaMdl,documents) returns the LDA topic indices with the largest probabilities for documents based on the LDA model ldaMdl.

MATLAB code: