EE69210: Machine Learning for Signal Processing Laboratory
Department of Electrical Engineering, Indian Institute of Technology, Kharagpur

Bayesian Classification

Anirvan Krishna | Roll no. 21EE38002

Keywords: Audio Classification, Bayesian Learning

Grading Rubric

	Tick the best applicable per row			**Points**
	Below Expectations	Lacking in Some	Meets all Expectations	
Completeness of the report				
Organization of the report (5 pts)				
Quality of figures (5 pts)				
Create train and test set using feature vectors (20 pts)				
Compute posterior probabilities (25 pts)				
Compute accuracy on the test set (20 pts)				
Compute the whole experiment on original signal vectors (25 pts)				
			TOTAL (100 pts)	

1. Theory: Feature extraction and bayes classifier

The Discrete Cosine Transform (DCT) is applied by dividing the input signal into small blocks and then performing the DCT on each block separately. For a signal s_0 of length D, the k-th DCT coefficient $x_0(k)$ is given by:

$$x_0(k) = \sqrt{\frac{2}{D}} \sum_{n=0}^{D-1} s_0[n] \cos\left(\frac{\pi k(2n+1)}{2D}\right), \quad 0 \leq k < D. \tag{1}$$

Let $\mathbf{x}_i$ be a column vector of dimension $D \times 1$ denoting the feature vector, consisting of real-valued numbers such that it is represented as $\mathbf{x}_i \in \mathbb{R}^{D\times 1}$:

$$\mathbf{x}_i = \begin{bmatrix} x(0) \\ x(1) \\ \vdots \\ x(D-1) \end{bmatrix}. \tag{2}$$

We further represent the set of all such N samples as a matrix $\mathbf{X}$ formed by stacking all the $\mathbf{x}_i$ in a column-wise manner such that $\mathbf{X} \in \mathbb{R}^{D\times N}$:

$$\mathbf{X} = [\mathbf{x}_0, \mathbf{x}_1, \ldots, \mathbf{x}_i, \ldots, \mathbf{x}_{N-1}]. \tag{3}$$

Explicitly, we can write:

$$\mathbf{X} = \begin{bmatrix} \mathbf{x}_0 & \mathbf{x}_1 & \ldots & \mathbf{x}_{N-1} \end{bmatrix}. \tag{4}$$

Similarly, let y_i be the label for $\mathbf{x}_i$ such that $\mathbf{Y} \in \mathbb{R}^{1\times N}$:

$$\mathbf{Y} = [y_0, y_1, \ldots, y_i, \ldots, y_{N-1}]. \tag{5}$$

We have K classes $C_1, C_2, \ldots, C_K$ and a set of n training examples $(\mathbf{x}_1, y_1), \ldots, (\mathbf{x}_n, y_n)$, where $\mathbf{x}_i \in \mathbb{R}^{d\times 1}$ is a d-dimensional feature vector and $y_i \in \{1, 2, \ldots, K\}$ is the corresponding class-label. Let $P(C_k)$ be the prior probability of class C_k, i.e., the probability that a randomly chosen example belongs to class C_k. Let $\mu_k \in \mathbb{R}^d$ and $\Sigma_k \in \mathbb{R}^{d\times d}$ be the mean vector and covariance matrix for class C_k. To classify a new example $\mathbf{x}$, we compute the posterior probability of each class C_k given $\mathbf{x}$ using Bayes' rule:

$$P(C_k|\mathbf{x}) = \frac{P(\mathbf{x}|C_k)P(C_k)}{P(\mathbf{x})}, \tag{6}$$

where

$$P(\mathbf{x}|C_k) = \frac{1}{(2\pi)^{d/2}|\Sigma_k|^{1/2}} \exp\left(-\frac{1}{2}(\mathbf{x}-\mu_k)^T \Sigma_k^{-1}(\mathbf{x}-\mu_k)\right). \tag{7}$$

Here, *mahalanobis distance* (d) is defined as:

$$d^2 = (\mathbf{x}-\mu_k)^T \Sigma_k^{-1}(\mathbf{x}-\mu_k) \tag{8}$$

Therefore, we can rewrite the posterior probability as follows:

$$P(C_k|\mathbf{x}) = \frac{\frac{1}{(2\pi)^{d/2}|\Sigma_k|^{1/2}} \exp\left(-\frac{1}{2}(\mathbf{x}-\mu_k)^T\Sigma_k^{-1}(\mathbf{x}-\mu_k)\right) P(C_k)}{\sum_{j=1}^{K} \frac{1}{(2\pi)^{d/2}|\Sigma_j|^{1/2}} \exp\left(-\frac{1}{2}(\mathbf{x}-\mu_j)^T\Sigma_j^{-1}(\mathbf{x}-\mu_j)\right) P(C_j)}. \tag{9}$$

Here, $|\Sigma_k|$ denotes the determinant of Σ_k, and Σ_k^{-1} is the inverse of Σ_k.
The class with the highest posterior probability is then selected as the predicted class for $\mathbf{x}$:

$$\hat{y} = \arg\max_{k\in\{1,2,\ldots,K\}} P(C_k|\mathbf{x}). \tag{10}$$

2. Data Preprocessing

The code preprocesses an audio file by performing the following steps:

(1) The audio file is loaded at a sampling rate of 44.1 kHz.
(2) It is split into two 30-second segments: $\mathbf{x}_v$ (first 30 seconds) and $\mathbf{x}_g$ (next 30 seconds).
(3) These segments are stacked into a matrix $\mathbf{X}$, with corresponding labels $\mathbf{y} = [0, 1]$ Here the label 0 corresponds to piano and the label 1 corresponds to violin.
(4) Each segment is further divided into frames of 20 ms, where the number of samples per frame is computed as:

$$\text{samples\_per\_frame} = sr \times 0.02. \tag{11}$$

(5) A segmentation model (`Segment`) is fitted to the data and then used to transform $\mathbf{X}$ and $\mathbf{y}$, resulting in the segmented dataset $\mathbf{s}_d$.

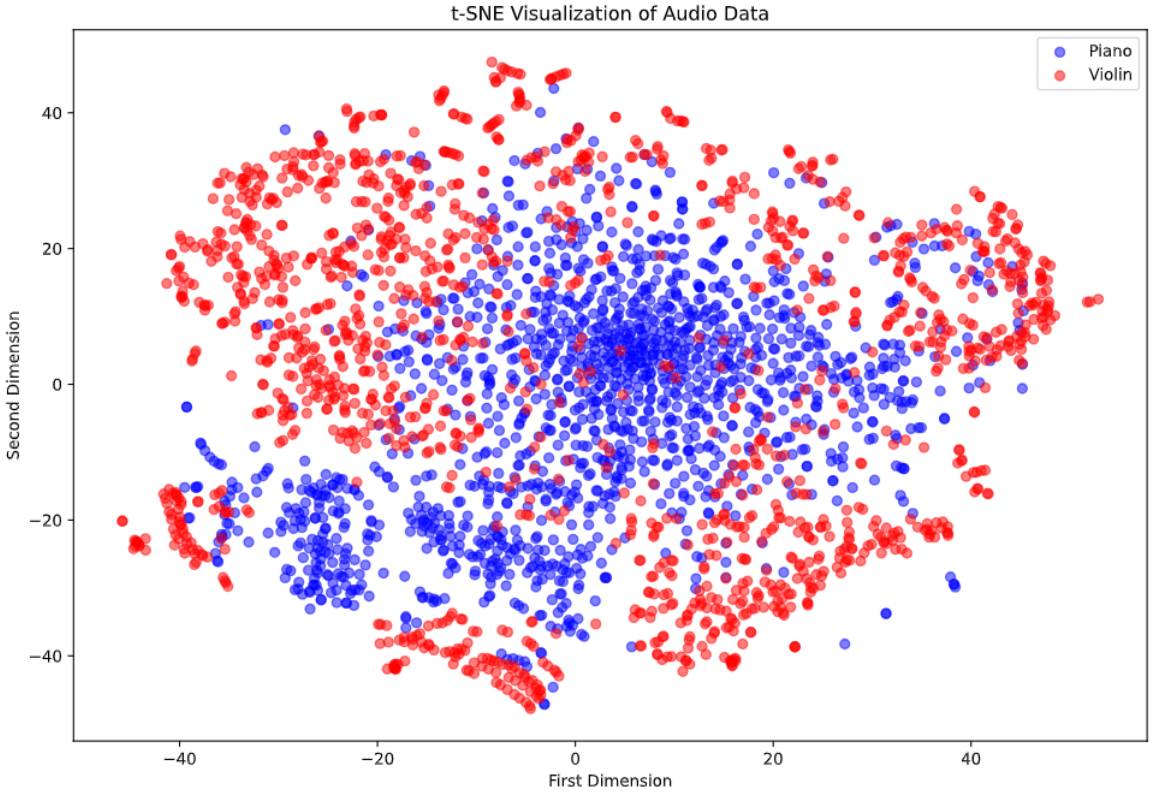

Fig. 1. t-SNE visualization of the audio data

After the above preprocessing, we extract the features from the data in form of DCT coefficients. For any signal of length K, we get K DCT coefficients for each frequency.

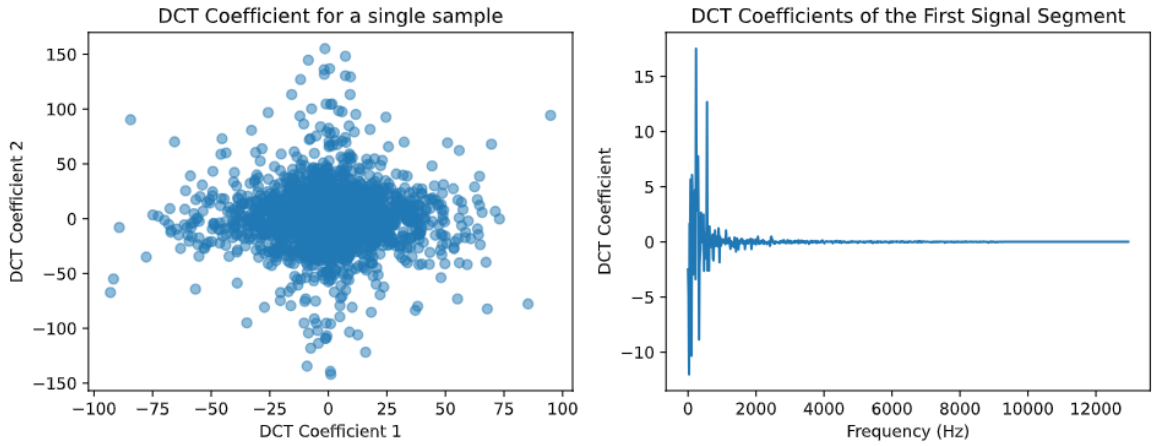

Fig. 2. Visualization of DCT coefficients of the data

In this case, the data consists of two classes `Piano` and `Violin`. We have $N = 3000$ audio samples. Therefore, label set $\omega \in \{0,1\}^N$. Each vector, $\mathbf{x}_i \in \mathbb{R}^{D\times 1}$. Here, each audio sample is of length 882 i.e., $D = 882$. We have the mean vector $\boldsymbol{\mu_0}, \boldsymbol{\mu_1} \in \mathbb{R}^D$ where $\boldsymbol{\mu_0}, \boldsymbol{\mu_1}$ are mean vectors corresponding to classes 0 and 1 respectively. Similarly, covariance matrices $\mathbf{R_0}, \mathbf{R_1} \in \mathbb{R}^{D\times D}$ where $\mathbf{R_0}, \mathbf{R_1}$ are covariance matrices corresponding to classes 0 and 1 respectively. We need to predict $p(\omega|\mathbf{x}_i)$. For the case, when the covariance matrices are singular, equation 7 remains no longer valid. So we use dimensionality reduction to handle this issue. We select 73 components out of 882 that preserve 99.02% of variance in the training data.

Algorithm 1 Handling Singular Covariance Matrices Using PCA

Input: Covariance matrices R_{x0}, R_{x1}, training data X_{train}, labels Y_{train}
Output: Transformed feature space if PCA is applied
Compute determinants:
$\det(R_{x0}) \leftarrow \text{determinant}(R_{x0})$, $\det(R_{x1}) \leftarrow \text{determinant}(R_{x1})$
Print determinants of R_{x0} and R_{x1}
if $\det(R_{x0}) = 0$ or $\det(R_{x1}) = 0$ **then**
    Request variance preservation ratio v_r from the user, where $0 < v_r \leq 1$
    Apply PCA:
    Fit PCA on X_{train} with variance preservation ratio v_r
    Transform X_{train} to obtain $X_{\text{train}}^{\text{PCA}}$
    Recompute Class Statistics:
    Extract transformed class-wise data:
    $X_0^{\text{PCA}} \leftarrow X_{\text{train}}^{\text{PCA}}$ where $Y_{\text{train}} = 0$, $X_1^{\text{PCA}} \leftarrow X_{\text{train}}^{\text{PCA}}$ where $Y_{\text{train}} = 1$
    Compute new means:
    $\mu_{x0}^{\text{PCA}} \leftarrow \text{mean}(X_0^{\text{PCA}})$, $\mu_{x1}^{\text{PCA}} \leftarrow \text{mean}(X_1^{\text{PCA}})$
    Compute new covariance matrices:
    $R_{x0}^{\text{PCA}} \leftarrow \text{cov}(X_0^{\text{PCA}})$, $R_{x1}^{\text{PCA}} \leftarrow \text{cov}(X_1^{\text{PCA}})$
end if

3. Inference results on data with DCT coefficients as features

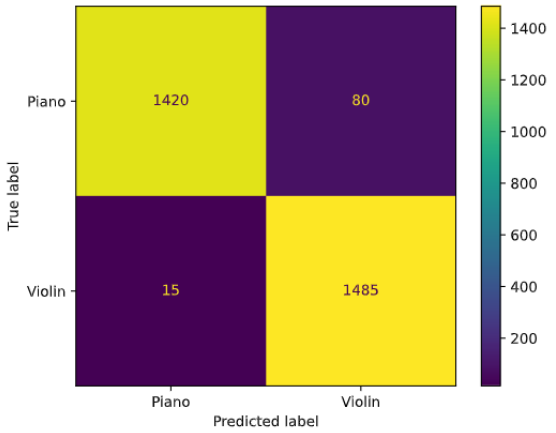

Fig. 3. Confusion matrix for training set

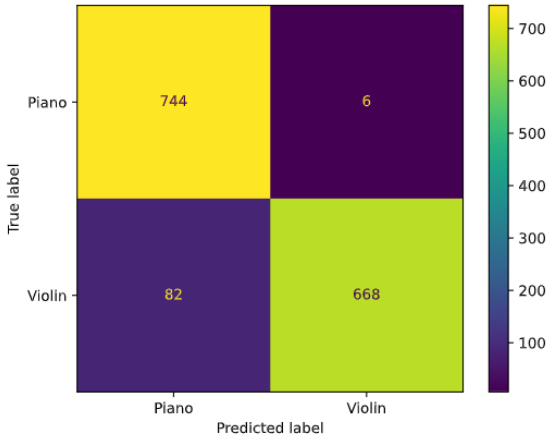

Fig. 4. Confusion matrix for test set

	Train Set			Test Set		
Class	**Precision**	**Recall**	**F1-Score**	**Precision**	**Recall**	**F1-Score**
0	0.99	0.95	0.97	0.90	0.99	0.94
1	0.95	0.99	0.97	0.99	0.89	0.94
Accuracy	0.97 (3000 samples)			0.94 (1500 samples)		
Macro Avg	0.97	0.97	0.97	0.95	0.94	0.94
Weighted Avg	0.97	0.97	0.97	0.95	0.94	0.94

Table 1. Classification Report for Train and Test Sets

4. Inference results on data with original signal features

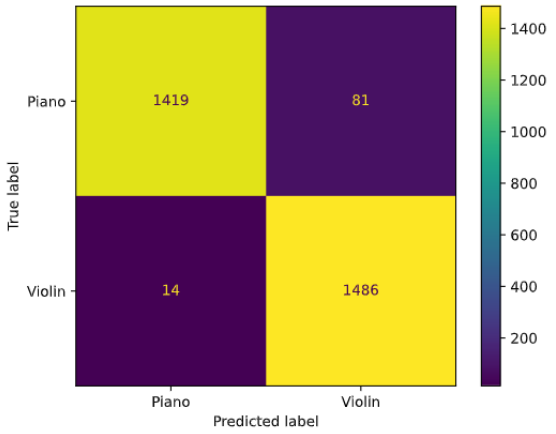

Fig. 5. Confusion matrix for training set

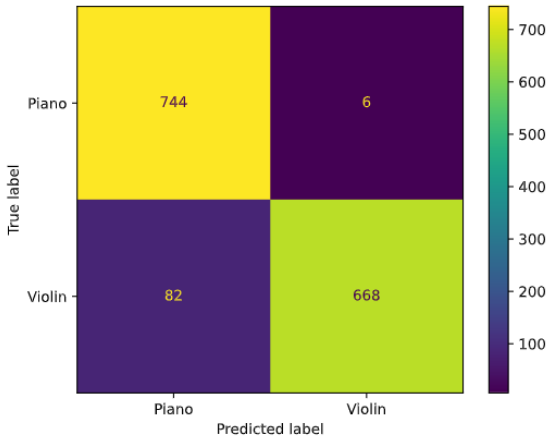

Fig. 6. Confusion matrix for test set

	Train Set			Test Set		
Class	**Precision**	**Recall**	**F1-Score**	**Precision**	**Recall**	**F1-Score**
0	0.99	0.95	0.97	0.90	0.99	0.94
1	0.95	0.99	0.97	0.99	0.89	0.94
Accuracy	0.97 (3000 samples)			0.94 (1500 samples)		
Macro Avg	0.97	0.97	0.97	0.95	0.94	0.94
Weighted Avg	0.97	0.97	0.97	0.95	0.94	0.94

Table 2. Classification Report for Train and Test Sets

When comparing the classification metrics after performing a Bayesian classification on raw signal features and features extracted after DCT, we observe a similar level of performance. We report an accuracy of 97% in the training set and 94% on the test set in both cases with almost similar confusion matrices. If we increase the amount of retained variance from 99% to 99.5%, we see that though the accuracy on the training set remains almost similar 97%, the accuracy on the test set increases to 95%. Therefore, we observe a correlation between the amount of variance retained during dimensionality reduction and the performance of the classifier. The more variance we retain, the better the classifier performs!