Department of Electrical Engineering
Indian Institute of Technology Kharagpur

Machine Learning for Signal Processing Laboratory
(EE69210)
Spring, 2022-23

Experiment 3: Bayes' Classifier

Grading Rubric

	Tick the best applicable per row			Points
	Below Expectation	Lacking in Some	Meets all Expectation	
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($\mathbf{s}_i$) (25 pts)				
			TOTAL (100 pts)	

1 Code of Conduct

Students are expected to behave ethically both in and out of the lab. Unethical behaviour includes, but is not limited to, the following:

- Possession of another person's laboratory solutions from the current or previous years.
- Reference to, or use of another person's laboratory solutions from the current or previous years.
- Submission of work that is not done by your laboratory group.
- Allowing another person to copy your laboratory solutions or work.
- Cheating on quizzes.

The rules of laboratory ethics are designed to facilitate these goals. We emphasize that laboratory TAs are available to help the student understand the basic concepts and answer the questions asked in the laboratory exercises. By performing the laboratories independently, students will likely learn more and improve their performance in the course as a whole. Please note that it is the student's responsibility to ensure that the content of their graded laboratories is not distributed to other students. If there is any question as to whether a given action might be considered unethical, please see the professor or the TA before you engage in such actions.

2 Introduction

2.1 Representation of signal as vector

Let $\{s[n]\} = \{s[0], s[1], ..., s[ND-1]\}$ represent an audio sequence, where ND is the number of digital samples in the audio. Let $\mathbf{s}_i \in \mathbb{R}^{D\times 1}$ be an alternate representation of this signal as column vector, such that there are no overlapping samples between the vectors.

$$\mathbf{s}_i = \begin{bmatrix} s[0] \\ s[1] \\ \vdots \\ s[D-1] \end{bmatrix} \tag{1}$$

We further represent the set of all such N vectors as a matrix $\mathbf{S} \in \mathbb{R}^{D\times N}$ formed by column-wise stacking of all the $\mathbf{s}_i$, so as to have

$$\mathbf{S} = \begin{bmatrix} \mathbf{s}_0, & \mathbf{s}_1, & \cdots, & \mathbf{s}_i, & \cdots, & \mathbf{s}_{N-1} \end{bmatrix} \tag{2}$$

$$\mathbf{S} = \begin{bmatrix} \begin{bmatrix} s[0] \\ s[1] \\ \vdots \\ s[D-1] \end{bmatrix}_0, & \begin{bmatrix} s[D] \\ s[D+1] \\ \vdots \\ s[2D-1] \end{bmatrix}_1, & \cdots, & \begin{bmatrix} s[iD] \\ s[iD+1] \\ \vdots \\ s[(i+1)D-1] \end{bmatrix}_i, & \cdots, & \begin{bmatrix} s[(N-1)D] \\ s[(N-1)D+1] \\ \vdots \\ s[ND-1] \end{bmatrix}_{N-1} \end{bmatrix} \tag{3}$$

here we denote y_i to be the label for $\mathbf{s}_i$ such that $\mathbf{Y} \in \mathbb{Z}^{1\times N}$.

$$\mathbf{Y} = \begin{bmatrix} y_0, & y_1, & \cdots, & y_i, & \cdots, & y_{N-1} \end{bmatrix} \tag{4}$$

2.2 Feature extraction of signal

Discrete Cosine Transform (DCT) is a mathematical technique for signal and image processing applications. It is widely used in various fields, including audio and video compression, digital watermarking, and computer vision. The DCT converts a signal from the time domain to the frequency domain. In other words, it takes a sequence of values, such as a digital audio or image signal. It transforms it into a set of coefficients representing different frequency components' contribution. This allows the signal to be compressed by discarding high-frequency components that may not be perceptually important.

The DCT is similar to the more well-known Fourier Transform (FT), which converts signals from the time domain to the frequency domain. However, the DCT uses only cosine functions, whereas the FT uses both sine and cosine functions. This makes the DCT more efficient for processing real-valued signals, like audio and images, because using only cosine functions results in a real-valued output. The DCT is applied by dividing the input signal into small blocks and then performing the DCT on each block separately. For a signal $\mathbf{s}_0$ of length D, the k-th DCT coefficient $x_0^{(k)}$ is given by:

$$x_0^{(k)} = \sqrt{\frac{2}{D}} \cdot \sum_{n=0}^{D-1} \mathbf{s}_0[n] \cdot \cos\left(\frac{\pi k(2n+1)}{2D}\right), \quad 0 \le k < D \tag{5}$$

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 $\mathbf{x}_i \in \mathbb{R}^{D\times 1}$.

$$\mathbf{x}_i = \begin{bmatrix} x^{(0)} \\ x^{(1)} \\ \vdots \\ x^{(D-1)} \end{bmatrix} \tag{6}$$

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 $\mathbf{X} \in \mathbb{R}^{D\times N}$.

$$\mathbf{X} = \begin{bmatrix} \mathbf{x}_0, & \mathbf{x}_1, & \cdots, & \mathbf{x}_i, & \cdots, & \mathbf{x}_{N-1} \end{bmatrix} \tag{7}$$

$$\mathbf{X} = \left[\begin{bmatrix} x^{(0)} \\ x^{(1)} \\ \vdots \\ x^{(D-1)} \end{bmatrix}_0, \begin{bmatrix} x^{(D)} \\ x^{(D+1)} \\ \vdots \\ x^{(2D-1)} \end{bmatrix}_1, \cdots, \begin{bmatrix} x^{(iD)} \\ x^{(iD+1)} \\ \vdots \\ x^{((i+1)D-1)} \end{bmatrix}_i, \cdots, \begin{bmatrix} x^{(N-1)D} \\ x^{((N-1)D+1)} \\ \vdots \\ x^{(ND-1)} \end{bmatrix}_{N-1} \right] \tag{8}$$

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

$$\mathbf{Y} = \begin{bmatrix} y_0, & y_1, & \cdots, & y_i, & \cdots, & y_{N-1} \end{bmatrix} \tag{9}$$

3 Bayes Classifier

Suppose 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 μ_k and $\mathbf{\Sigma}_k$ 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:

$$P(C_k|\mathbf{x}) = \frac{P(\mathbf{x}|C_k)P(C_k)}{P(\mathbf{x})} = \frac{\frac{1}{(2\pi)^{d/2}|\mathbf{\Sigma}_k|^{1/2}} \exp\left(-\frac{1}{2}(\mathbf{x}-\mu_k)^T\mathbf{\Sigma}_k^{-1}(\mathbf{x}-\mu_k)\right) P(C_k)}{\sum_{j=1}^{K} \frac{1}{(2\pi)^{d/2}|\mathbf{\Sigma}_j|^{1/2}} \exp\left(-\frac{1}{2}(\mathbf{x}-\mu_j)^T\mathbf{\Sigma}_j^{-1}(\mathbf{x}-\mu_j)\right) P(C_j)}, \quad (10)$$

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

$$\hat{y} = \underset{k\in 1,2,\ldots,K}{\text{argmax}}\ P(C_k|\mathbf{x}).$$

4 Feature Extraction

4.1 Load the WAV files

Let sample_audio be the name of the audio file to be loaded. We can load the audio file and resample it to a sampling rate of 44100 Hz using the librosa.load function:

```python
import IPython.display as display
import librosa
import librosa.display
import numpy as np
from seglearn.transform import Segment
from scipy.fft import dct, idct

sample_audio = 'the_godfather_theme_instru_mix.wav'
x, sr = librosa.load(sample_audio, sr=44100)
```

Listing 1: Loading the audio file

where, x is a one-dimensional NumPy array containing the audio signal, and sr is the signal's sampling rate (in Hz).

4.2 Data Preparation

1. The following code provided slicing a 1-dimensional NumPy array x into two segments, x_v : the first 30 seconds of x, x_g: the 30-60 seconds of x

```python
x_v = x[:sr*30]
x_g = x[sr*30:sr*60]
X=np.stack((x_v,x_g))
```

Listing 2: Slicing the sample audio

The first line of code uses the numpy.stack() function to vertically stack the arrays x_v and x_g, creating a 2-dimensional array X. The resulting array X has two rows (corresponding to the x_v and x_g segments) and $sr \times 30$ columns (the length of each segment).
2. The following code creates a new NumPy array from the Python list [0, 1]. The resulting NumPy array y is a 1-dimensional array with two elements, 0 at index 0 and 1 at index 1. Since y has only two elements, it can be considered a binary vector or a binary classification label, where 0 and 1 represent the two possible classes.

```python
y = np.array([0,1])
```

Listing 3: Labels for two classes

3. The following code calculates the number of audio samples in a single frame of 0.02 seconds, given the audio signal's sampling rate sr.

```python
samples_per_frame=int(sr*0.02)
```

Listing 4: number of samples in a single frame

Frame-based audio processing is a standard audio signal processing and analysis technique. Dividing the audio signal into frames allows us to analyze the signal's spectral and temporal characteristics over shorter intervals, which can reveal information about changes in the signal over time, such as the onset and decay of individual sound events.

4.3 Non overlapping segment

The code uses the Segment class from the seglearn package to segment the audio signal X into fixed-width frames, each labeled by the corresponding class label y.

```
ts = Segment(width=samples_per_frame,overlap=0).fit(X,y)
s_d = ts.transform(X,y)
```

Listing 5: Segment audio signal

4.4 Compute DCT

The code uses the dct function from the scipy.fft module to compute the discrete cosine transform (DCT) of the segmented audio data s_d.

```
X_d=dct(s_d[0])
```

Listing 6: compute DCT

dct(s_d[0]) applies the DCT to the segmented audio data along the second axis (i.e., the samples axis), producing a 3-dimensional NumPy array of the same shape as s_d[0]. The resulting X_d array contains the DCT coefficients for each frame of the segmented audio data.

4.5 Training Data

The following code splits the DCT coefficients X_d and their corresponding labels s_d[1] into training data and labels and randomly shuffles them.

```
x_train, y_train = X_d, s_d[1]
from sklearn.utils import shuffle
x_train, y_train = shuffle(x_train, y_train, random_state=0)
```

Listing 7: training data

5 Objectives

5.1 Assignments to solve and report

Write down the codes for the above tasks in the .ipynb file, and submit the executed file. Submit a separate pdf describing your observations and reasoning while executing these experiments.

1. Segment the input sequence $\{s[n]\}$ into N vectors. Show the size of each vector.
2. Compute the DCT on each D dimensional vector and show size of the generated feature vector
3. Visualize the generated dataset on 2D space.
4. Compute the mean vector and covariance matrix for each class. Show the size of the mean vector and the covariance matrix.
5. Compute the likelihood function $P(\mathbf{x}|C_k)$ for each class k, compute $P(\mathbf{x})$, and compute the prior probability $P(C_k)$

6. Create the test set using the audio file provided and show the dimension of each feature vector.
7. Compute the posterior probability for each test sample and compute the classification accuracy on the test set.
8. Also, design the Bayes classifier using the original signal vectors($\mathbf{s}_i$) and compute the classification accuracy.