Queen Mary
University of London

ECS708 Machine Learning

Assignment 2: Clustering and MoG

Aim

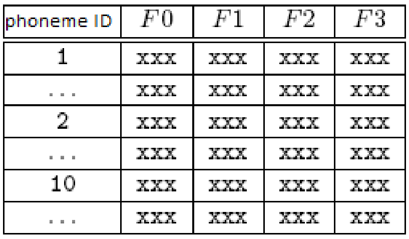
The aim of this assignment is to become familiar with clustering using the Mixture of Gaussians model.

1. Introduction

For this lab, we will use the Peterson and Barney's dataset of vowel formant frequencies. (For more info, look at http://speech:ucsd:edu/aldebaro/papers/klautau02_pbvowel.pdf. – a copy of this article is at QMplus)

More specifically, Peterson and Barney measured the fundamental frequency F0 and the first three formant frequencies (F1-F3) of sustained English Vowels, using samples from various speakers.

The dataset can be found in the QMplus, in the files of Assignment 2, in the folder named "data", at the file "PB_data.npy". Load the file. In your workspace, you will have 4 vectors (f0-f3), containing the fundamental frequencies (F0, F1, F2 and F3) for each phoneme and another vector "phoneme_id" containing a number representing the id of the phoneme. The arrangement of the data is as follows:



phoneme ID	$F0$	$F1$	$F2$	$F3$
1	xxx	xxx	xxx	xxx
...	xxx	xxx	xxx	xxx
2	xxx	xxx	xxx	xxx
...	xxx	xxx	xxx	xxx
10	xxx	xxx	xxx	xxx
...	xxx	xxx	xxx	xxx

In the exercises that follow, we will use only the dataset associated with formants F1 and F2.

2. MoG Modelling using the EM Algorithm

Recall the following definition of a Mixture of Gaussians. Assuming our observed random vector is x, a MoG models p(x) as a sum of weighted Gaussians. More specifically:

$$p(x) = \sum_{k=1}^{K} \frac{p(c_k)}{(2\pi)^{\frac{D}{2}} det(|\Sigma_k|)^{\frac{1}{2}}} \exp\left(-\frac{1}{2}(x-\mu_k)^T \Sigma_k^{-1}(x-\mu_k)\right)$$

Where D is the dimension of vector x, μ_k, Σ_k and $p(c_k)$ are the mean vector, covariance matrix and the weight of the k-th gaussian, and K is the number of the gaussians used.

Task 1:
Load the dataset to your workspace. We will only use the dataset for F1 and F2, arranged into a 2D matrix where the first column will be F1 and the second column will be F2. Using the code in **task_1.py**, produce a plot of F1 against F2. (You should be able to spot some clusters already in this scatter plot.).

Include in your report the corresponding lines of your code and the plot. **[20 points]**

Task 2:
Train the data for phonemes 1 and 2 with MoGs. You are provided with python files **task_2.py, get_predictions.py** and **plot_gaussians.py**.
Specifically, you are required to:

1. Look at the **task_2.py** and **get_predictions.py** code files and understand what they are calculating. Pay particular attention to the initialisation of the means and covariances (also note that in this Task, the code is only estimating diagonal covariances).
2. Generate a dataset stored in the variable "X_phoneme_1", that contains only the F1 and F2 for the first phoneme.
3. Run **task_2.py** on the dataset using K=3 Gaussians (run the code a number of times and note the differences.) Save your MoG model: this should comprise the variables mu, s and p.
4. Run **task_2.py** on the dataset using K=6
5. Repeat steps 2-4 for the second phoneme

Include in your report the lines of code you wrote, and results that illustrate the learnt models. **[20 points]**

Task 3:
Use the 2 MoGs (K=3) learnt in Task 2 to build a classifier to discriminate between phonemes 1 and 2. Classify using the Maximum Likelihood (ML) criterion (feel free to hack parts from the code in **task_3.py**, and utilize the **get_predictions.py** file so that you calculate the likelihood of a data vector for each of the two MoG models). Calculate the mis-classification error. Remember that a classification under the ML compares $p(x;\theta_1)$, where $\theta_1$are the parameters of the MoG learnt for the first phoneme, with $p(x;\theta_2)$, where θ_2 are the parameters of the MoG learnt for the second phoneme.

Repeat this for K=6 and compare the results.

Include in your report the lines of the code that your wrote, explanations of what the code does and comment on the differences on the classification performance **[20 points]**

Task 4:
For this Task, you will use the code file **task_4.py**. Create a grid of points that spans the two datasets (i.e., the dataset that contains phoneme 1, and the dataset that contains phoneme 2). Classify each point in the grid using one of your classifiers (i.e., the MoG that was trained on phoneme 1 and the MoG that was trained on phoneme 2). That is, create a classification matrix, M, whose elements are either 0 or 1. M(i,j) is 0 if the point x=[x1(i), x2(j)] is classified as belonging to phoneme 1, and is 1 otherwise. x1 is a vector whose elements are between the minimum and the maximum value of F1 for the first two phonemes, and x2 similarly for F2.

Display the classification matrix.
Include the lines of code in your report, comment them, and display the classification matrix. **[20 points]**

Task 5:
In the code of **task_5.py** a MoG with full covariance matrices is fit to the data. Now, create a new dataset that will contain 3 columns, as follows:
X = [F1, F2, F1+F2]

Fit a MoG model to the new data. What is the problem that you observe? Explain why.

Suggest ways of overcoming the singularity problem and implement them.

Include the lines of code in your report, and graphs/plots so as to support your observations. **[20 points]**

Write a report about what you have done along with relevant graphs. Save the solution in a folder with your ID.
Create and submit:

1) **a .zip that contains all of your code for both parts and**
2) **a copy of your report. The report should be in .pdf format named as ML_assignment2-StudentID.pdf**