[Assignment 4: Gaussian Mixture Models](https://e.centennialcollege.ca/d2l/lms/dropbox/user/folder_submit_files.d2l?db=664299&grpid=0&isprv=0&bp=0&ou=1010603" \o "Submit files to Assignment 4: Gaussian Mixture Models)

DINH HOANG VIET PHUONG – 301123263

1. **Libraries Import:**

Importing essential libraries: numpy for numerical operations, fetch\_olivetti\_faces to load the dataset, PCA for dimensionality reduction, GaussianMixture for the clustering model, matplotlib.pyplot for plotting, and rotate from skimage.transform to rotate images.

A screenshot of a computer

Description automatically generated

1. **Loading Dataset:**

Fetching the Olivetti faces dataset with shuffling enabled and a fixed random state for reproducibility. Storing images, data, and targets in separate variables

A screen shot of a computer code

Description automatically generated

Initializing a PCA object to preserve 99% of the variance. whiten=True helps in making the data unit-variance. Applying PCA on the data and then printing the original and reduced number of features.

The original Olivetti faces dataset images are of size 64x64 pixels, leading to a total of 64×64=4096 features (each pixel is a feature). After applying PCA to retain 99% of the variance, the feature space was reduced from 4096 dimensions down to 260 dimensions. This is a substantial reduction and can help in speeding up the subsequent clustering using the GMM.

1. **Best GMM Configuration**

A close-up of a computer screen

Description automatically generated

Setting Up Covariance Types:

The BIC (Bayesian Information Criterion) is used to determine the best covariance type for the GMM. The BIC is a measure that penalizes models based on their complexity. Lower BIC values are preferred.

\_Defining possible covariance types for GMM: Four types of covariance matrices (full, tied, diag, spherical) are tested, and the one with the lowest BIC is chosen.

\_Initializing variables for the best covariance type determination.A computer screen shot of a code

Description automatically generated

Iterating through each covariance type, initializing a GMM with 40 components as an example, setting the covariance type, and fitting the reduced data.

Calculating the Bayesian Information Criterion (BIC) and Akaike Information Criterion (AIC) for each covariance type.

Storing the values in respective lists and determining the best covariance type based on the lowest BIC.

1. **Determine the optimal number of clusters for the best covariance type**

A screenshot of a computer code

Description automatically generated

Setup for Optimal Number of Clusters:  
\_Setting a range for the number of components/clusters from 1 to 50.  
\_Initializing a list to store BIC values for each number of clusters.

Determining Optimal Clusters:  
\_Iterating through each possible number of clusters, initializing a GMM with the previously determined best covariance type, fitting the data, and calculating the BIC.

Finding Optimal Number:  
\_Finding the number of components that results in the lowest BIC.

1. **Plot the results**

A screenshot of a computer code

Description automatically generated

Plotting BIC values for different covariance types and the number of components to visualize the best choices.

A graph and a graph

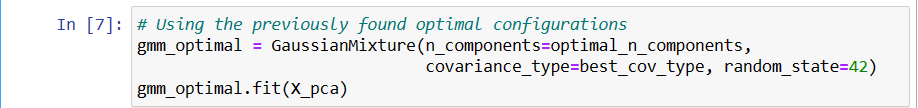
Description automatically generated with medium confidence



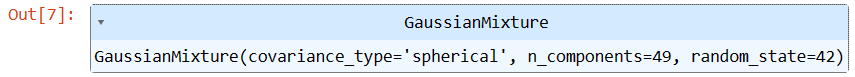
Displaying the determined best covariance type and optimal number of clusters.  
Best covariance type according to BIC: spherical

Optimal number of components according to BIC: 49

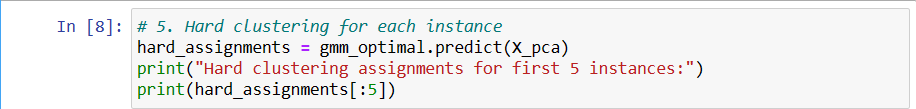
1. **Using the previously found optimal configurations**



Initializing and fitting an optimal GMM based on the best covariance type and number of clusters found earlier.



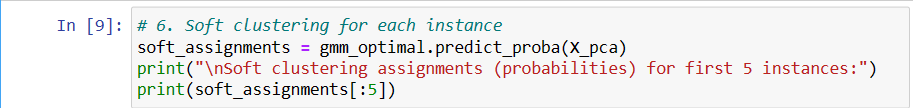
1. **Hard clustering for each instance**



Making hard assignments for each instance (assigning each instance to the most probable cluster) and displaying the assignments for the first 5 instances:



1. **Soft clustering for each instance**



Making soft assignments (getting the probability of each instance belonging to each cluster) and displaying the results for the first 5 instances.

A close up of a grid

Description automatically generated

1. **Generate new faces using the sample() method**

A screenshot of a computer code

Description automatically generated

Generating new faces using the GMM and visualizing them

A close-up of a person's face

Description automatically generated

1. **Modify some images (e.g., rotate, flip, darken)**

Taking a sample face, applying different modifications (rotation, flipping, darkening), and visualizing the results.

A screenshot of a computer program

Description automatically generated

A close-up of a person's face

Description automatically generated

1. **Scoring Modified Images**

A screenshot of a computer program

Description automatically generated

Getting scores for original and modified images to understand how likely these images are under the GMM's learned distribution.

A black text on a white background

Description automatically generated

The score for the original images is approximately -354.70. This score is a measure of how well the data fits the model. The lower the score (more negative), the better the fit, as this score is often based on negative log-likelihood.

The rotated image has a score of approximately -500.22, which is worse than the original image score. This suggests that the rotated image is less consistent with the model than the original images.

The flipped image score is around -337.16, which is better than the original score. This might mean that the flipped image is more like the patterns captured by the model than the original ones.

The darkened image has a score very close to the original, at approximately -354.11. This suggests that darkening the image doesn't drastically change its representation or relationship with the underlying model.