Practical Machine Learning – Project 2
Unsupervised Learning

Stan Eduard George - Gr. 407

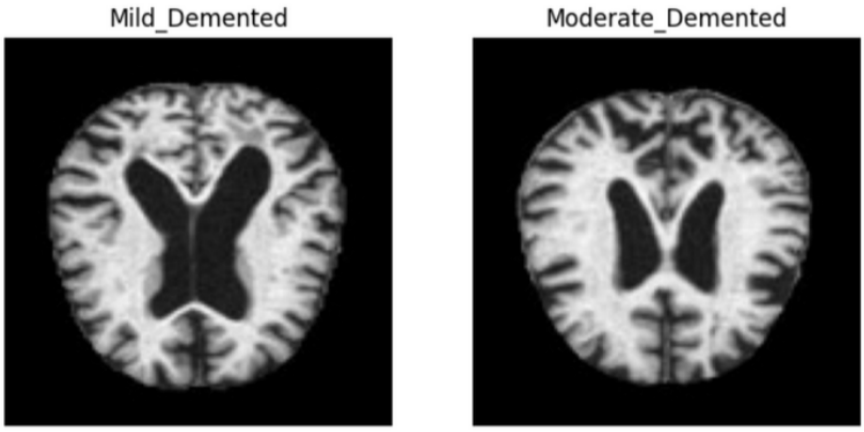
1 Dataset

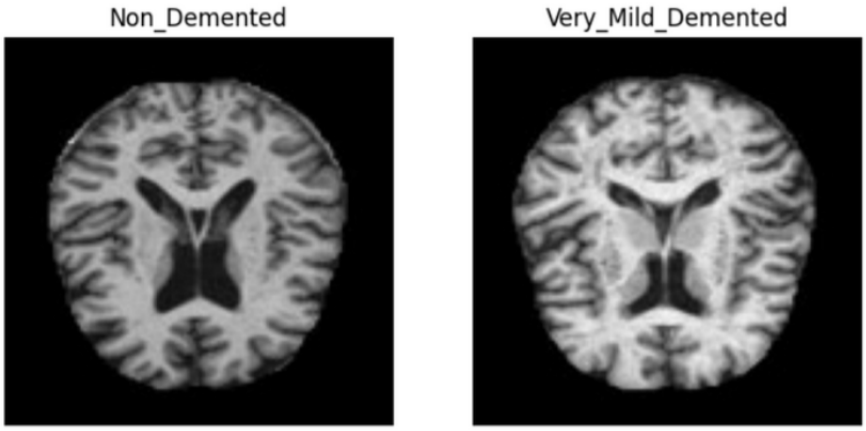
The dataset chosen for this project is Alzheimer MRI. Basically, there are 6400 images containing preprocessed MRI (Magnetic Resonance Imaging) images of 128x128 pixels. Each image corresponds to one of these four classes:

- Class - 1: Mild Demented (896 images)
- Class - 2: Moderate Demented (64 images)
- Class - 3: Non Demented (3200 images)
- Class - 4: Very Mild Demented (2240 images)

2 Preprocessing

The code starts with loading the images and labels into numpy arrays. The extraction is executed with the help of cv2 (opencv) that converts the raw images into RGB format. After that, I plotted an image for each category:





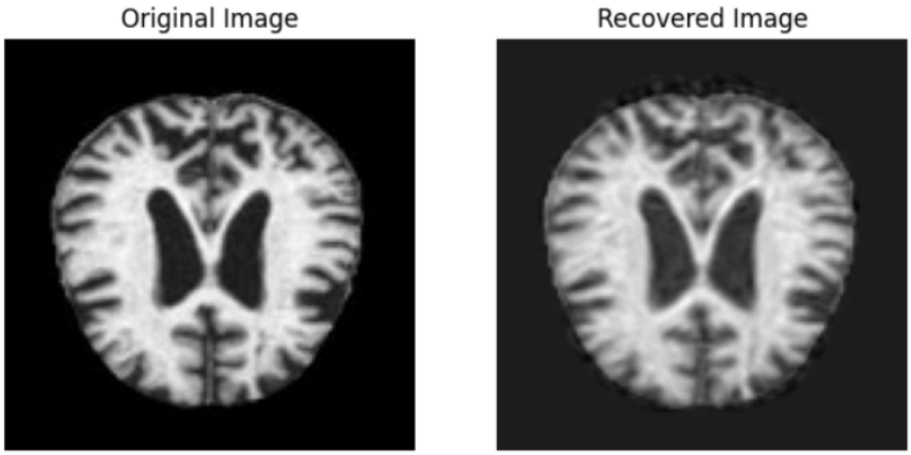
Because the images are black and white, I decided to remove the RGB dimension and keep them gray. This way, the number of dimensions is divided by 3.

3 The first set of features

For the first set of features, I decided to use Principal Component Analysis (PCA). To understand the particularities of this approach, we need to look inside the black box of PCA:

1. Centers the data by extracting the mean of each feature.
2. Calculates the covariance matrix (the relationship between each feature).
3. Computes eigenvalues and eigenvectors of the covariance matrix (eigenvalues can be interpreted as the magnitude of variance, and the eigenvectors as the directions of the maximum variance).
4. Principal components are chosen based on the amount of variance we want to retain in the dataset.
5. The original data is projected onto this new space (the centered data matrix is multiplied by the projection matrix).

The first thing I did here is to fit an empty PCA on the gray images. After that, I compute the cumulative sum of explained_variance_ratio_ parameter of the model (which orders the variances from the highest to the lowest). Then, I choose the percentage of explained variance to be left and I compute the number of features to be used further. I can easily do that because the total variance is already normed, and I just need to select the points where the cumulative variance is higher than a certain value. After we end up with the number of features with the highest variance, we fit another PCA over the data, using the number of components equal to the resulted number. Here is the representation of an image that was decompressed from 95% variance, and the original image.



So, the first set of features is represented by the images compressed to 5% variance using PCA. This data was also normed.

4 The second set of features

The second set is generated using a Resnet18 convolutional neural network. The features are extracted from the last layer, GlobalAveragePooling2D(), which collapses the spatial dimensions to a single value per channel. More details about the implementation are shown in the code.

5 The metrics

The metrics we choose to use to evaluate the model are:

- Silhouette score - indicates how well-separated the clusters are. It ranges from -1 to 1. A high value indicates that the point is well matched with its cluster and poorly matched with the other clusters.
- Fowlkes-Mallows Index (FMI) - evaluates the similarity between the true labels and the predicted ones. It ranges from 0 to 1 and measures the geometric mean of precision and recall.
- Adjusted Rand Index (ARI) - measures the similarity between two clustering results (true and predicted labels in our case). It specifies the similarity with random chance, ranging from -1 to 1, 1 meaning perfect agreement with the true labels, and 0 meaning that the predicted labels are similar to random chance.

6 Unsupervised models

6.1 HDBSCAN

Hierarchical Density-Based Spatial Clustering of Applications with Noise (HDBSCAN)

- Identifies clusters based on regions of high data point density.
- Builds a hierarchy of clusters, organizing them into a tree structure (dendrogram).

- As the algorithm progresses, it forms clusters by merging or splitting existing clusters.
- The hierarchy is constructed by considering different density thresholds.
- For example, as the density threshold increases, larger clusters are created.

6.1.1 Results with PCA

min_cluster_size - minimum number of data points required to form a cluster

Table 1: Grid search for min_cluster - PCA

min_cluster	unique_labels	silhouette	F-M Score	Adj. Rand Score
param-not-set	219	-0.1268	0.1356	0.0153
1	242	-0.1384	0.1066	0.0141
3	242	-0.1384	0.1066	0.0141
5	242	-0.1384	0.1066	0.0141
10	242	-0.1384	0.1066	0.0141
50	242	-0.1384	0.1066	0.0141

max_cluster_size - maximum number of data points required to form a cluster.
The following results are extracted with min_cluster_size parameter has no value.

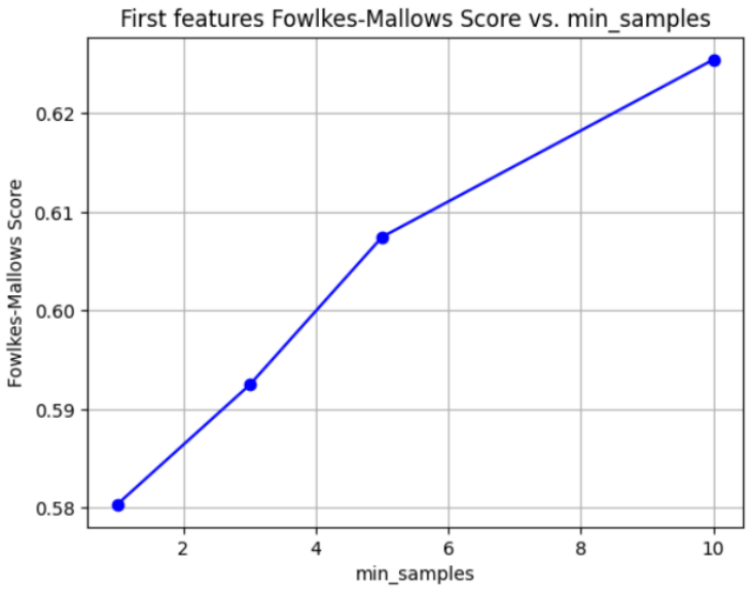
Table 2: Grid search for max_cluster - PCA

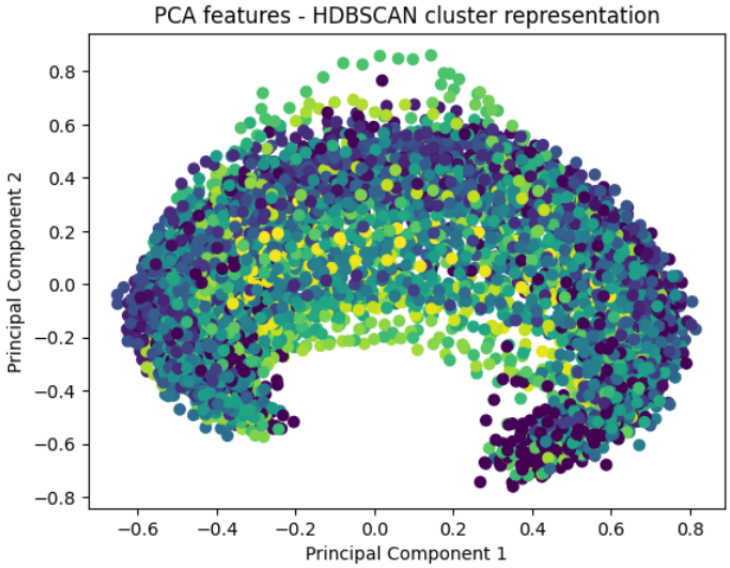
max_cluster	unique_labels	silhouette	F-M Score	Adj. Rand Score
1 - exception	-	-	-	-
3 - exception	-	-	-	-
4 - exception	-	-	-	-
5	40	-0.2501	0.6133	0.0127
10	142	-0.2086	0.5420	0.0202
50	219	-0.1268	0.1356	0.0153

min_samples - influences the density sensitivity.

Table 3: Grid search for min_samples - PCA

min_samples	**unique_labels**	**silhouette**	**F-M Score**	**Adj. Rand Score**
1	100	-0.2483	0.5804	0.0052
3	73	-0.2598	0.5925	0.0030
5	41	-0.2635	0.6075	0.0017
10	3	0.1046	0.6254	0.0004
12 - exception	-	-	-	-
20 - exception	-	-	-	-





6.1.2 Results with Resnet18

Table 4: Grid search for min_cluster_size - Resnet18

min_cluster_size	unique_labels	silhouette_score	fowlkes_mallows_score
1 - exception	-	-	-
3	720	-	-
4	4	-0.0324	0.6234
5	3	0.0904	0.6231
10	3	0.0897	0.6226
20	3	-0.0955	0.4786

We stick with min_cluster_size=4.

Table 5: Grid search for max_cluster_size - Resnet18

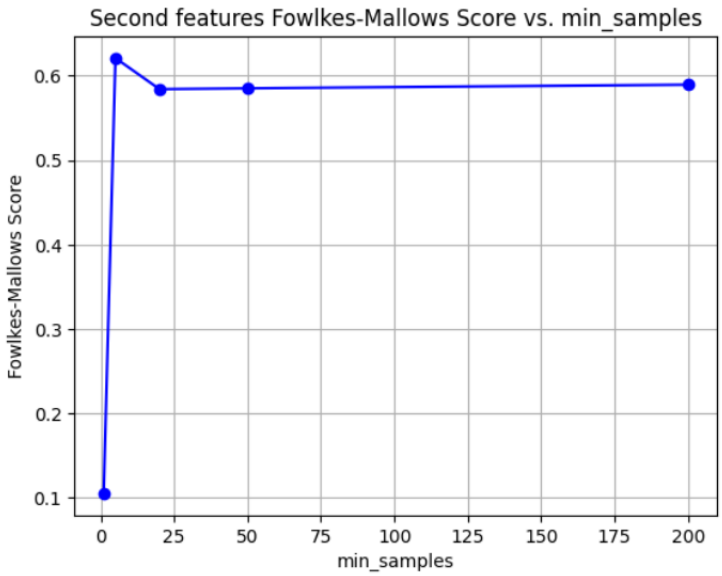
max_cluster_size	unique_labels	silhouette_score	fowlkes_mallows_score
4	113	-0.4760	0.5798
5	174	-0.4502	0.5510

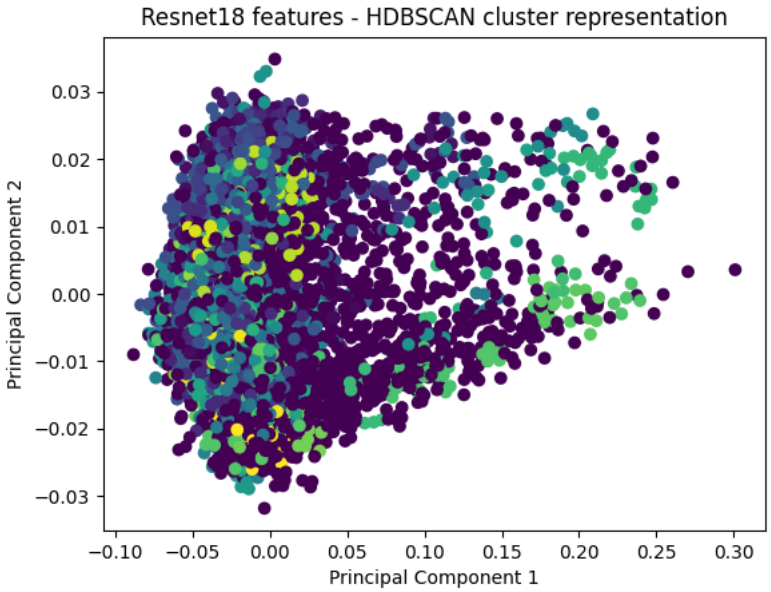
10	370	-0.3140	0.4137
3200	415	-0.2003	0.2831

Table 6: Grid search for min_samples - Resnet18

min_samples	**unique_labels**	**silhouette_score**	**fowlkes_mallows_score**
1	713	-0.0231	0.1053
5	3	0.0199	0.6209
20	7	-0.3055	0.5841
200	3	-0.2156	0.5893
500	1	-	-

adjusted_rand_score for the final approach (hdb = HDBSCAN(min_cluster_size=4)) is 0.006874.





6.2 GMM

Gaussian Mixture Model.

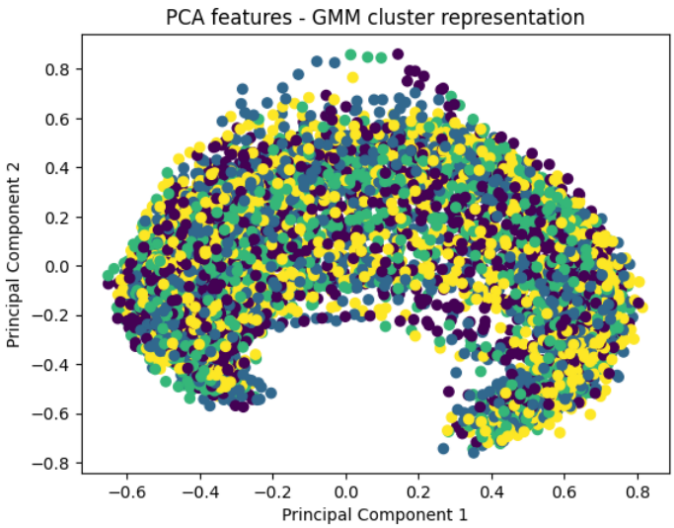
- Used for representing a mixture of multiple Gaussian distributions.
- It assumes that the data points are generated from a mixture of several Gaussian distributions, each associated with a cluster.

The hyperparameters are:

- n_components=4 - specifies the number of Gaussian distributions that will be used to model the dataset.
- init_params='kmeans' - specifies how the initial parameters of the model should be initialized before the fitting process.
- max_iter=1000 - the maximum number of iterations.
- tol=1e-3 - convergence criterion.

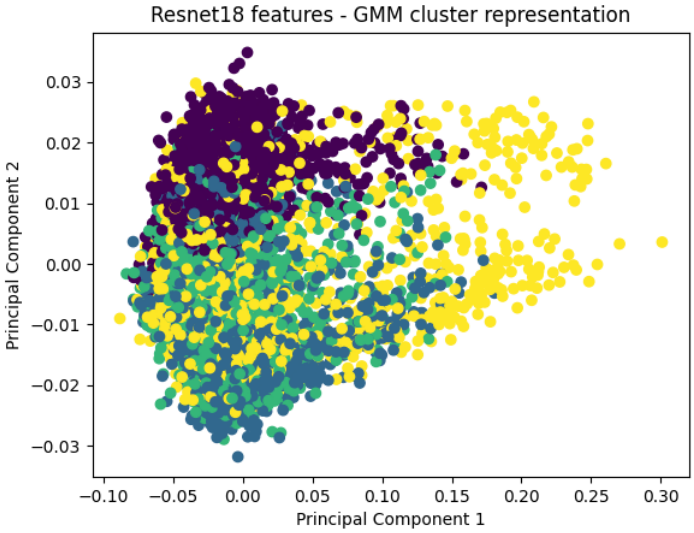
6.2.1 Results with PCA

silhouette_score	fowlkes_mallows_score	adjusted_rand_score
-0.0067	0.3139	0.0011



6.2.2 Results with Resnet18

init_params	unique_labels	silhouette	fowlkes_mallows	adjusted_rand
'kmeans'	4	0.0099	0.3279	0.0362
'random'	4	-0.0766	0.3172	0.0351



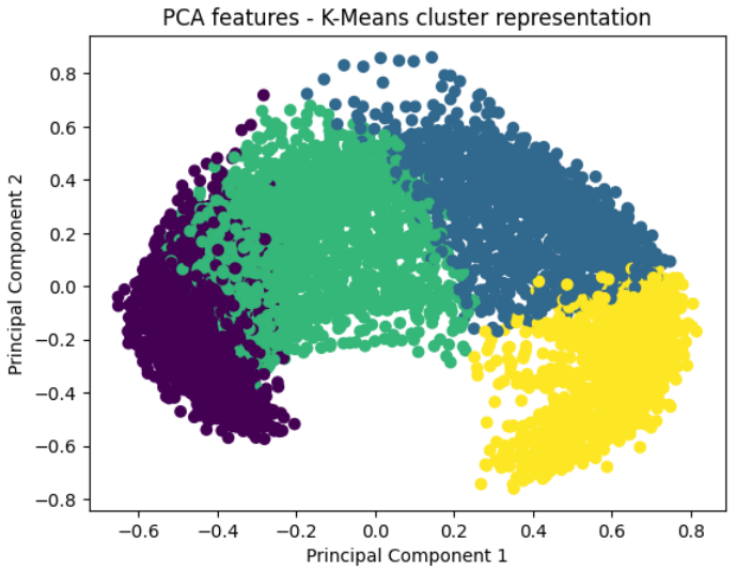
6.3 K-Means

The hyperparameters used are:

- n_clusters=4 – specifies the number of clusters that the algorithm should aim to identify
- n_init=10 - the number of times the algorithm will be run with different initializations of centroids

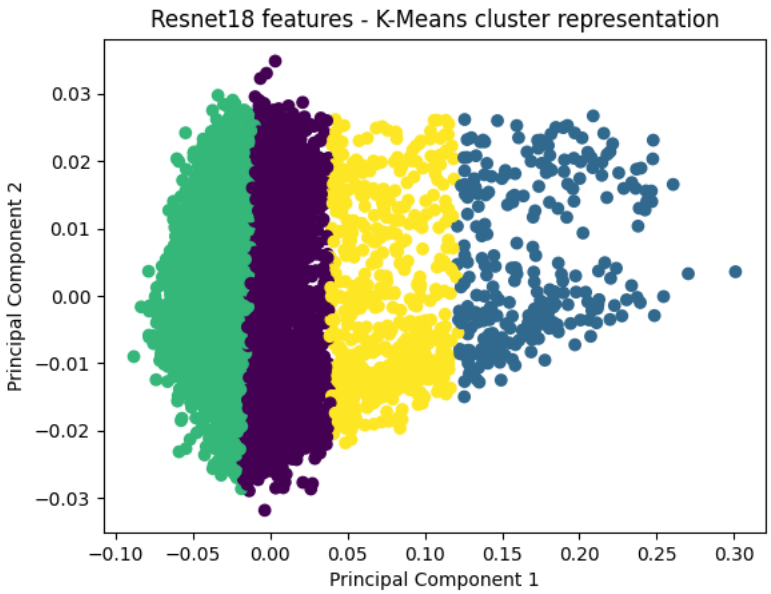
6.3.1 Results with PCA

silhouette_score	fowlkes_mallows_score	adjusted_rand_score
0.1230	0.3232	0.0044



6.3.2 Results with Resnet18

silhouette_score	fowlkes_mallows_score	adjusted_rand_score
0.1956	0.4052	0.0448



7 Comparison with a supervised model

The supervised model I used is Random Forest Classifier. No hyperparameters were tuned for this model.

7.1 Results with PCA

Class	Precision	Recall	F1-Score	Support
Mild_Demented	1.00	0.53	0.70	243
Moderate_Demented	1.00	0.28	0.43	18
Non_Demented	0.82	1.00	0.90	773
Very_Mild_Demented	0.94	0.88	0.91	566
Accuracy			0.88	1600
Macro Avg	0.94	0.67	0.74	1600
Weighted Avg	0.89	0.88	0.87	1600

7.2 Results with Resnet18

Class	Precision	Recall	F1-Score	Support
Mild_Demented	0.93	0.35	0.51	220
Moderate_Demented	0.00	0.00	0.00	5
Non_Demented	0.77	0.92	0.84	806
Very_Mild_Demented	0.73	0.72	0.72	569
Accuracy			0.76	1600
Macro Avg	0.61	0.50	0.52	1600
Weighted Avg	0.78	0.76	0.75	1600

Note: After I've filtered the results generated by Resnet18 using PCA of 97% variance, the results increased by a small amount. Here is the classification report:

Class	Precision	Recall	F1-Score	Support
Mild_Demented	1.00	0.40	0.58	218
Moderate_Demented	0.00	0.00	0.00	17
Non_Demented	0.78	0.98	0.86	809
Very_Mild_Demented	0.80	0.71	0.75	556
Accuracy			0.80	1600
Macro Avg	0.64	0.52	0.55	1600
Weighted Avg	0.81	0.80	0.78	1600

8 Another way to compare with random choice

For this last approach, I chose DummyClassifier from Scikit-Learn. The only parameter modified is strategy='uniform', which generates predictions uniformly at random from the list of unique labels observed in y_train. The results for both PCA and Resnet18 features are shown in the following table:

Class	Precision	Recall	F1-Score	Support
Mild_Demented	0.13	0.24	0.17	226
Moderate_Demented	0.00	0.06	0.01	18
Non_Demented	0.49	0.26	0.34	780
Very_Mild_Demented	0.37	0.26	0.30	576
Accuracy			0.25	1600
Macro Avg	0.25	0.20	0.20	1600
Weighted Avg	0.39	0.25	0.30	1600