Dedole Tommy
https://github.com/D-Tommy/IBM-Course

Pair-reviewed project :
Tumor detection using MRI dataset & images

Dataset Used

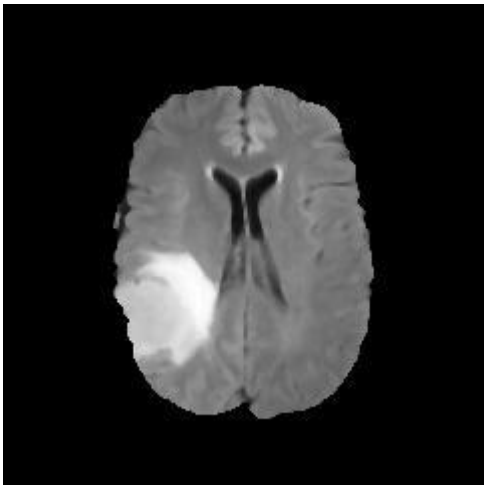
API : kaggle datasets download -d jakeshbohaju/brain-tumor

direct link : Brain Tumor

Content

This is a brain tumor feature dataset including five first-order features and eight texture features with the target level (in the column Class : 1 = tumor, 0 = no tumor.).

- First Order Features
  - Mean
  - Variance
  - Standard Deviation
  - Skewness
  - Kurtosis
- Second Order Features
  - Contrast
  - Energy
  - ASM (Angular second moment)
  - Entropy
  - Homogeneity
  - Dissimilarity
  - Correlation
  - Coarseness



The dataset contains 3762 images associated with 3762 entries in the dataset with 56% / 44% of class repartition, respectively absence and presence of tumor.

Objective

Use clustering models in order to detect brain tumor based on datasets containing features extracted from images as well as on images themselves.

- In this case, the focus will be made clustering rather than on dimensionality reduction, even though PCA is used before using cluster algorithms on images.

Code used for this project is available on my github

1) Data cleaning and exploration

	count	mean	std	min	25%	50%	75%	max
Class	3762.0	4.473684e-01	0.497288	0.000000e+00	0.000000e+00	0.000000e+00	1.000000e+00	1.000000e+00
Mean	3762.0	9.488890e+00	5.728022	7.865906e-02	4.982395e+00	8.477531e+00	1.321272e+01	3.323997e+01
Variance	3762.0	7.111011e+02	467.466896	3.145628e+00	3.632255e+02	6.225804e+02	9.669543e+02	2.910582e+03
Standard Deviation	3762.0	2.518227e+01	8.773526	1.773592e+00	1.905847e+01	2.495156e+01	3.109589e+01	5.394981e+01
Entropy	3762.0	7.360262e-02	0.070269	8.815796e-04	6.856456e-03	6.662805e-02	1.132844e-01	3.945386e-01
Skewness	3762.0	4.102727e+00	2.560940	1.886014e+00	2.620203e+00	3.422210e+00	4.651737e+00	3.693129e+01
Kurtosis	3762.0	2.438907e+01	56.434747	3.942402e+00	7.252852e+00	1.235909e+01	2.264030e+01	1.371640e+03
Contrast	3762.0	1.279615e+02	109.499601	3.194733e+00	7.212521e+01	1.067374e+02	1.610590e+02	3.382574e+03
Energy	3762.0	2.047051e-01	0.129352	2.473117e-02	6.961745e-02	2.254965e-01	2.989014e-01	5.896818e-01
ASM	3762.0	5.863159e-02	0.058300	6.116308e-04	4.846590e-03	5.084867e-02	8.934206e-02	3.477246e-01
Homogeneity	3762.0	4.792519e-01	0.127929	1.054898e-01	3.649727e-01	5.125512e-01	5.755566e-01	8.109208e-01
Dissimilarity	3762.0	4.698498e+00	1.850173	6.811207e-01	3.412363e+00	4.482404e+00	5.723821e+00	2.782775e+01
Correlation	3762.0	9.557667e-01	0.026157	5.494262e-01	9.471379e-01	9.616098e-01	9.713547e-01	9.899724e-01
Coarseness	3762.0	7.458341e-155	0.000000	7.458341e-155	7.458341e-155	7.458341e-155	7.458341e-155	7.458341e-155

Immediately : ‘Coarseness’ is a column composed of a unique same value, it is then dropped.

1.1) Dealing with skewed data.

	skew
Mean	0.773241
Variance	1.071127
Standard Deviation	0.184992
Entropy	0.954313
Skewness	4.332738
Kurtosis	11.308936
Contrast	10.249119
Energy	0.185695
ASM	1.102947
Homogeneity	-0.287805
Dissimilarity	1.954792
Correlation	-4.753668

Keeping the most skewed column →

	skew
Mean	0.773241
Variance	1.071127
Entropy	0.954313
Skewness	4.332738
Kurtosis	11.308936
Contrast	10.249119
ASM	1.102947
Dissimilarity	1.954792

Applying log transform to these →

	skew
Mean	-0.595351
Variance	-1.099685
Standard Deviation	0.184992
Entropy	0.816068
Skewness	1.425495
Kurtosis	1.296765
Contrast	-0.369895
Energy	0.185695
ASM	0.973880
Homogeneity	-0.287805
Dissimilarity	0.004298
Correlation	-4.753668

Outliers are then removed from the ‘**Kurtosis**’ feature, 82 rows are dropped. Points are considered as outliers if they are not contained in :

$I = [q25\% - (\frac{3}{2} * IQR),\ q75\% + (\frac{3}{2} * IQR)]$ with IQR = q75% - q25%

'Correlation' feature is still skewed to the right, a Boxcox transformation is applied to this column.

	skew
Class	0.255420
Mean	-0.348773
Variance	-0.787788
Standard Deviation	0.258232
Entropy	0.784882
Skewness	0.748555
Kurtosis	0.702697
Contrast	-0.544469
Energy	0.150402
ASM	0.942869
Homogeneity	-0.284823
Dissimilarity	-0.201752
Correlation	-0.141663
Kmeans	0.346527
AggClust	0.320466

The dataset is now almost not skewed, and ready to be scaled.

1.2) Scaling.

Since clustering algorithms that use distances metrics will be used, the whole dataset is standardly scaled.

```
from sklearn.preprocessing import StandardScaler
sds = StandardScaler()
```

```
X = df.drop(['Class', 'Image'], axis = 1)
y = df['Class']
```

```
X_scaled = sds.fit_transform(X)
```

The dataset is now properly usable.

2) Clustering on dataset

Since this dataset is made to detect tumors and not classify them, the target feature only presents 2 values. Clustering algorithms will therefore be trained in order to retrieve these 2 clusters. Since the study aims here to detect a pathology, recall will be the most important metrics to look for. Indeed, if a tumor is not detected, it can develop and the patient's condition will deteriorate quickly, while if it is detected complementary diagnosis can be runned to confirm its presence.

2.1) K-Means

Fitting the whole processed dataset with K = 2, KMeans yielded the following results :

Kmeans	Class	number
0	0	35
	1	1491
1	0	2038
	1	116

	precision	recall	f1-score	support
0	0.98	0.95	0.96	2154
1	0.93	0.98	0.95	1526
accuracy			0.96	3680
macro avg	0.96	0.96	0.96	3680
weighted avg	0.96	0.96	0.96	3680

It appears that with 98% recall, there's only 2% of tumors that are undetected.

2.2) Agglomerative Clustering (ward)

Fitting the whole processed dataset, until 2 clusters remain, using ward linkage and euclidean distance metrics, agglomerative clustering yielded the following results :

Class	AggClust	number
0	0	2052
	1	21
1	0	79
	1	1528

	precision	recall	f1-score	support
0	0.99	0.96	0.98	2131
1	0.95	0.99	0.97	1549
accuracy			0.97	3680
macro avg	0.97	0.97	0.97	3680
weighted avg	0.97	0.97	0.97	3680

In this case, agglomerative clustering outperformed the K-means algorithm.

The two previous algorithms applied to the dataset offered high quality results under the form of tremendous recall score, meaning that in this case clustering algorithms are really efficient at detecting tumors based on graphical features extracted from images. The idea is now to study if they can perform as well if directly used on images rather than on features.

3) Clustering applied to MRI images

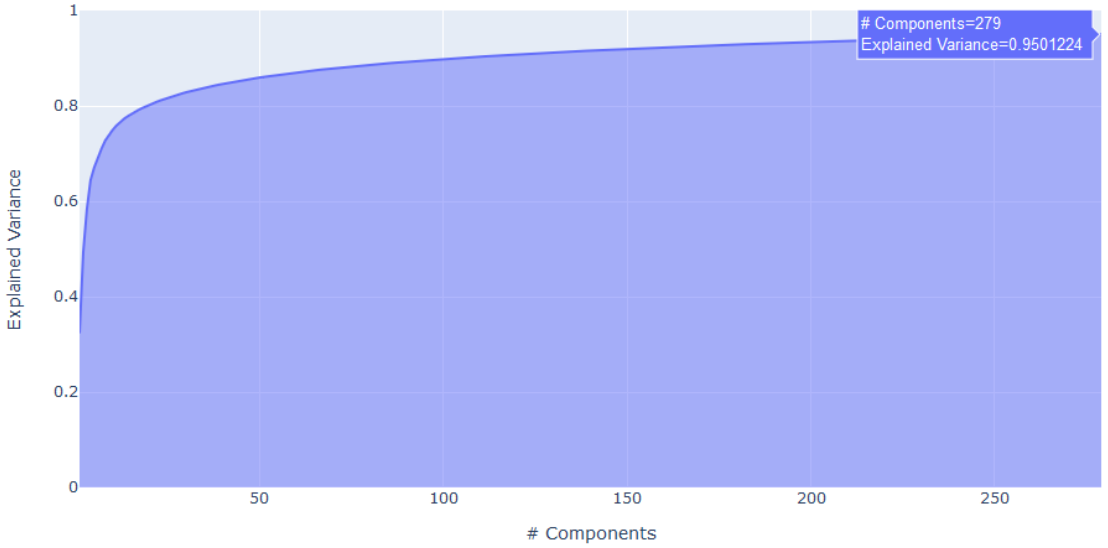
Results obtained on other brain MRI images are provided in annexes. The case presented here serves as an explicit example.

3.1) Image Pre-processing

Provided images are 240x240 pixels in grayscale. They're resized to 64x64 px.
The 3762 images are then loaded and flattened in a list of 4096 elements, resulting in a dataset of 3762 rows and 4096 columns.

Since the images are not heavily blurred there's no need for noise reduction algorithms in this case.

Then, a PCA performed on the whole dataset to extract the N firsts eigenvectors that will explain 95% (arbitrarily chosen) of the variance of the data.
This allow one to reduce the dimension of the data from 4096 to 279 (i.e : the 279 firsts eigenvectors explains 95% of the variance)

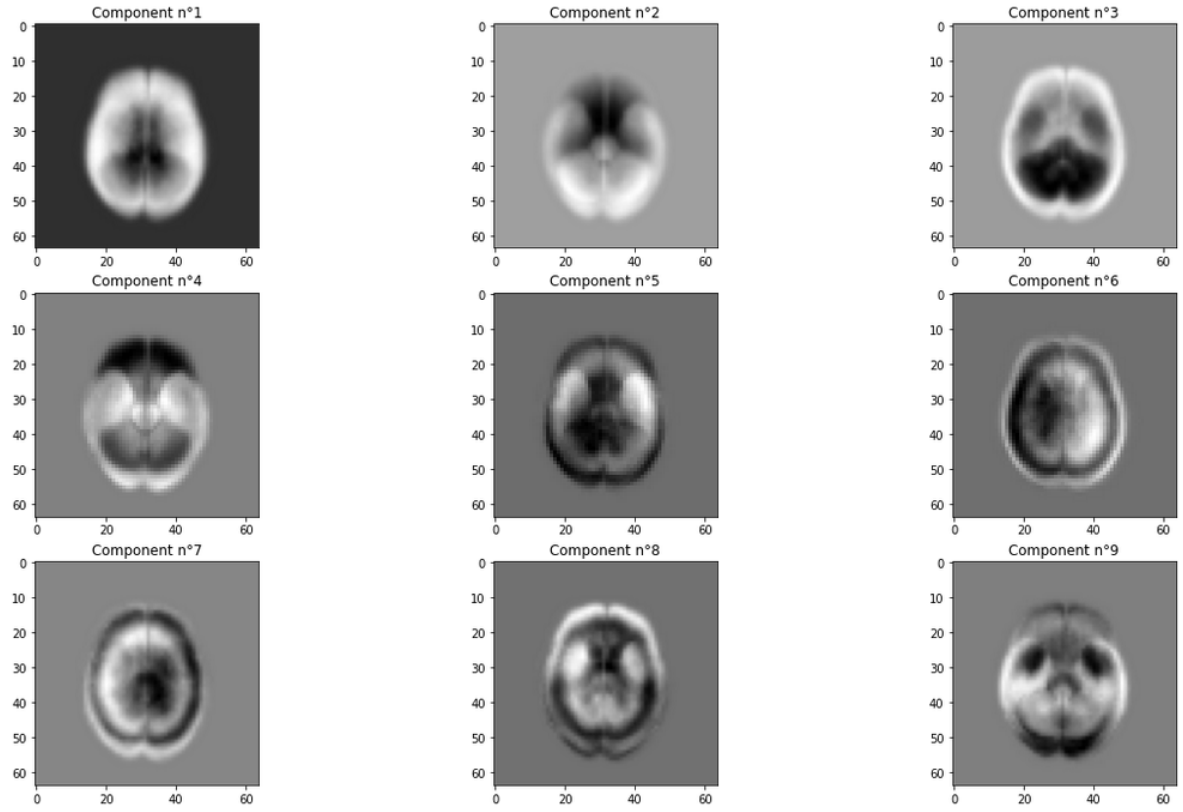


Then, a projection matrix is build :

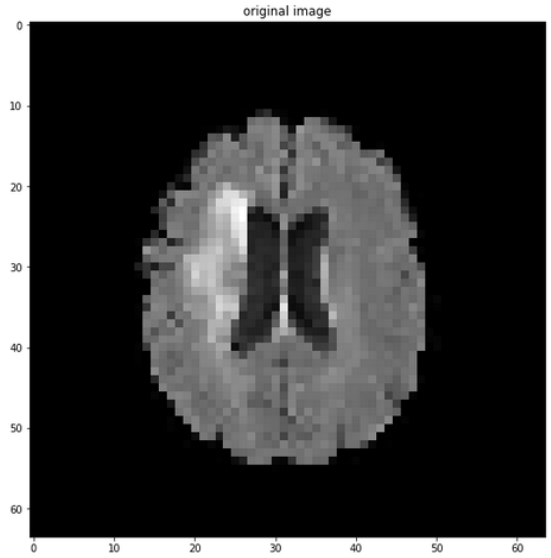
[30]:
```
#Creating projection matrix
required_dim = pca.components_.shape[0]
proj_matrix = np.empty(shape =(X.shape[1], required_dim))

for index in range(required_dim):
    proj_matrix[:,index] = pca.components_[index]
print(f'projection matrix shape : {proj_matrix.shape}')
```

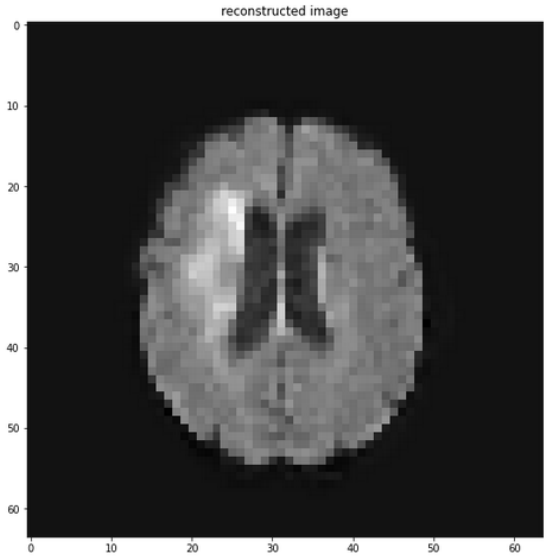
```
projection matrix shape : (4096, 279)
```

This matrix contains the 279 eigenvectors in flattened lists. These lasts are accessible and can provide one with insight of what visuals characteristics are the most important in the whole image dataset :



Using these components (eigenvectors) it is possible to reconstruct the original images with significantly lower dimensionality :

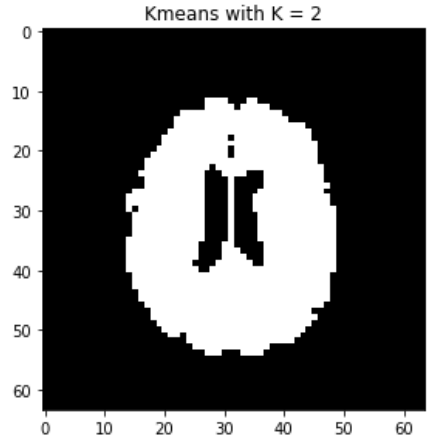


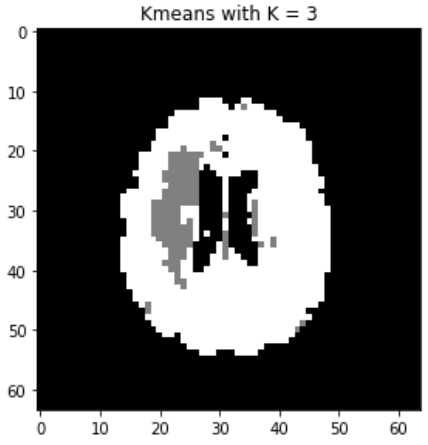


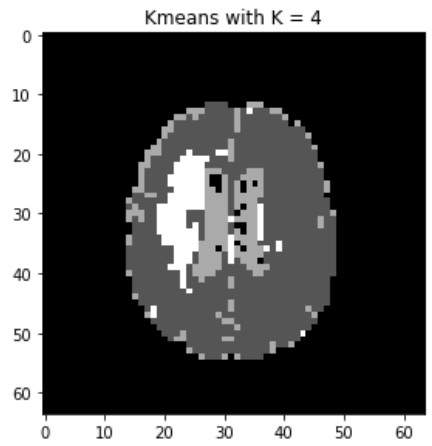
3.2) Clustering on MRI images

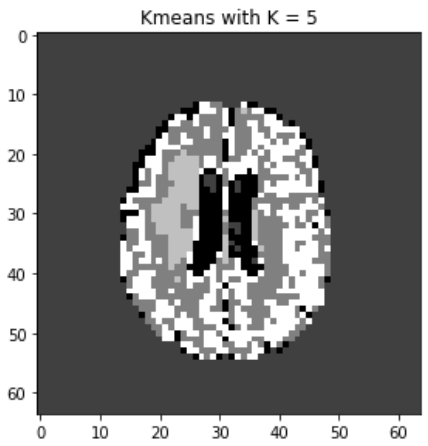
3.2.1) K-Means

A K-mean algorithm is trained with few different K on the reconstructed images :





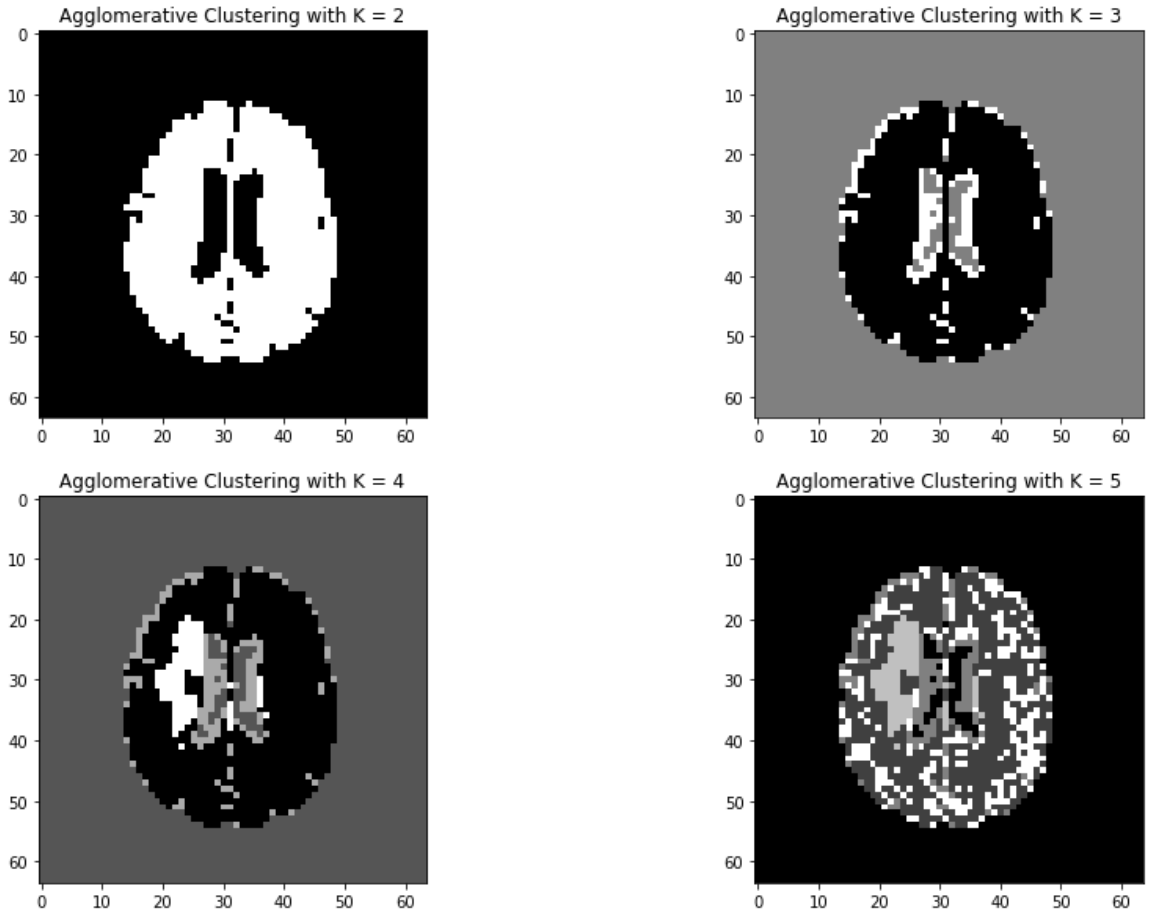




The tumor is already clearly visible when K = 3. Once $K \geq 5$the results start to be less and less interpretable.

3.2.2) Agglomerative Clustering

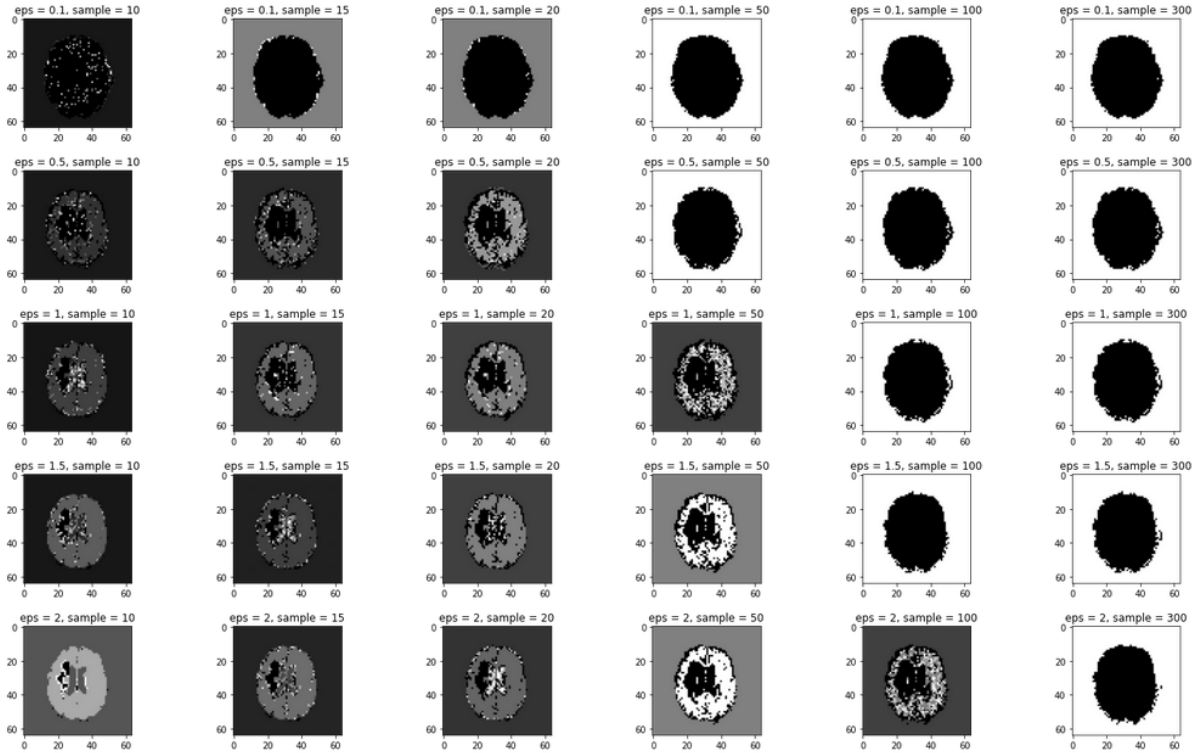
Same principle as for K-Mean, an agglomerative clustering algorithm is trained with several K and results are displayed below :



This time, the tumor appears as a cluster from K = 4.

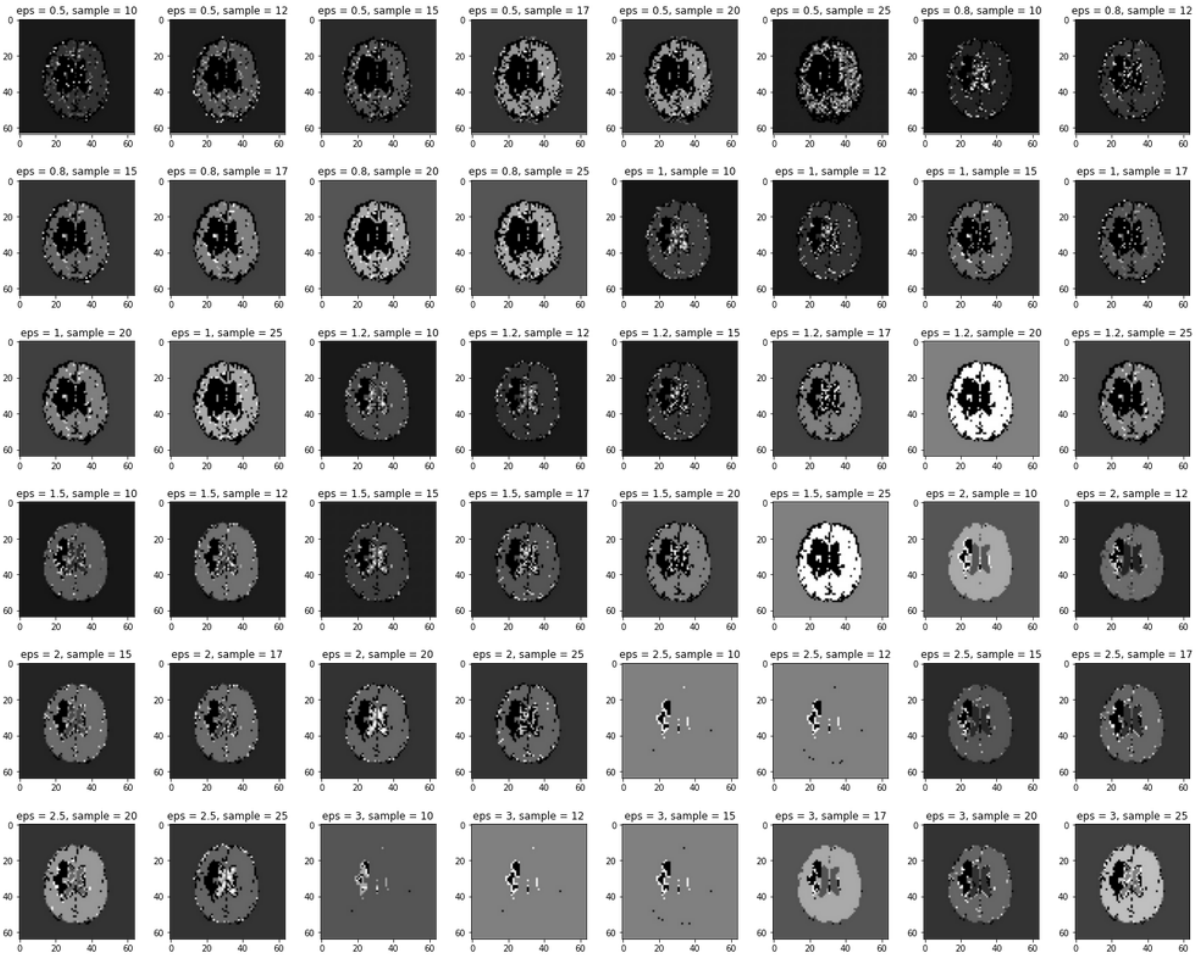
3.3.3) Density-Based Spatial Clustering of Applications with Noise (DBSCAN)

In opposition with Kmeans and agglomerative clustering, DBSCAN requires a fine tuning of hyperparameters to define the density threshold for core member definition. a first search over a relatively wide scale of parameters is performed :



Epsilon (maximum distance for a point to be considered as a neighbor) varies between 0.1 and 2 while the minimum of sample varies between 10 and 300.

Using results obtained by the first search, a second one is performed to tune further the two hyperparameters :



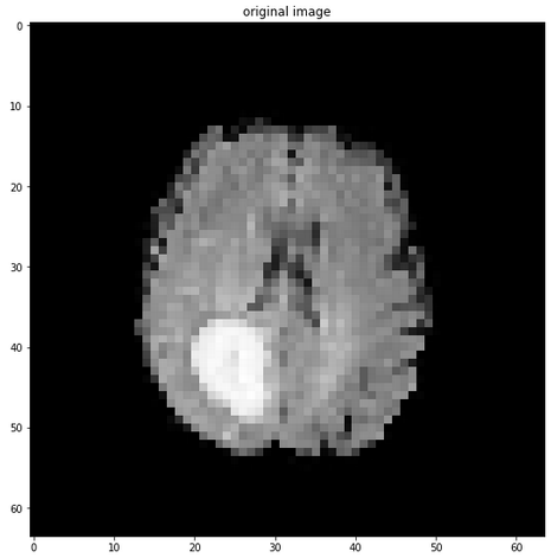
Some parameters even allow the extraction of the tumor itself, which is an impressive result.

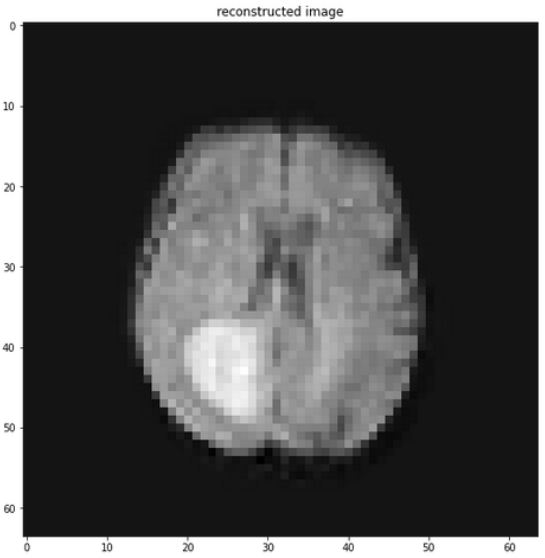
Conclusion

Clustering algorithms yield qualitative results whereas they are used to perform classification tasks on extracted features dataset or directly on MRI image after a PCA dimensionality reduction.
Results obtained using DBSCAN are deeply interesting when used with the right parameters and could benefit from a deeper study. The first point that could be invested in my opinion is building a pipeline of image processing comparing extracted clusters from DBSCAN to standardized pictures of the brain with tumors and then build an AI to decide if it is indeed an anomaly or not.

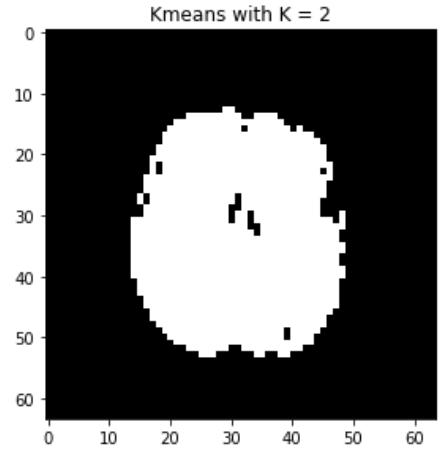
Annexe 1 : MRI image with tumor 2

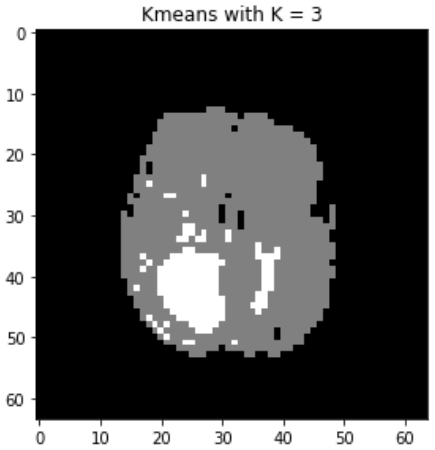
PCA Reconstruction :

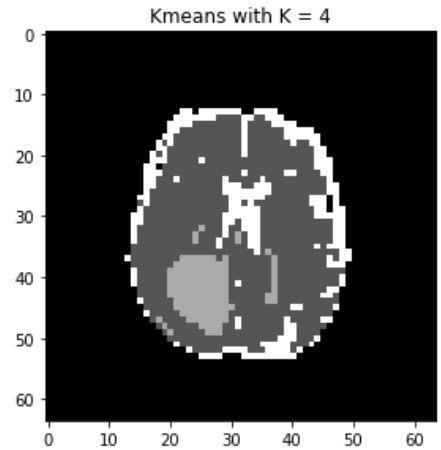


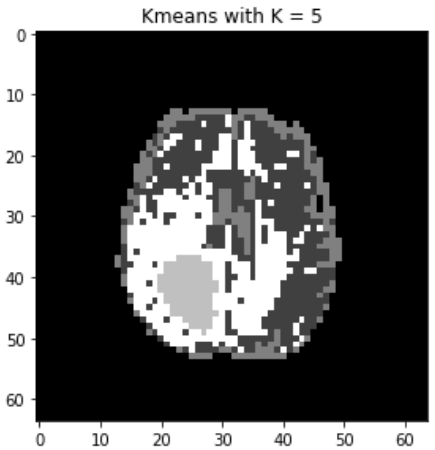


Kmeans :

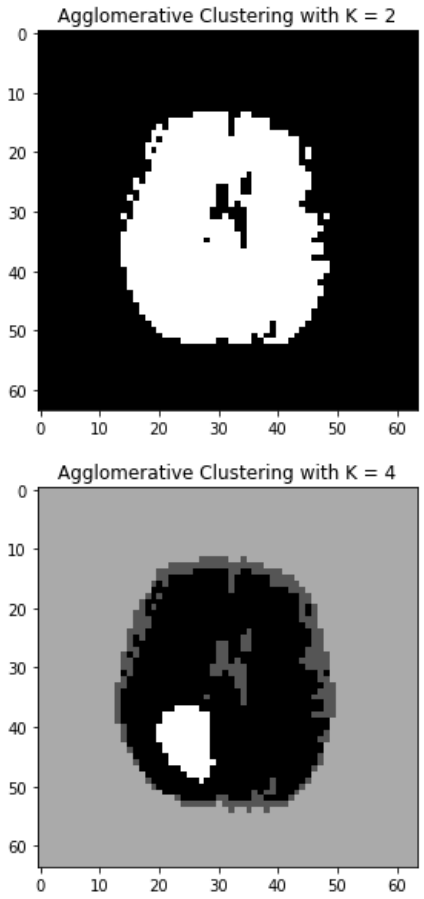






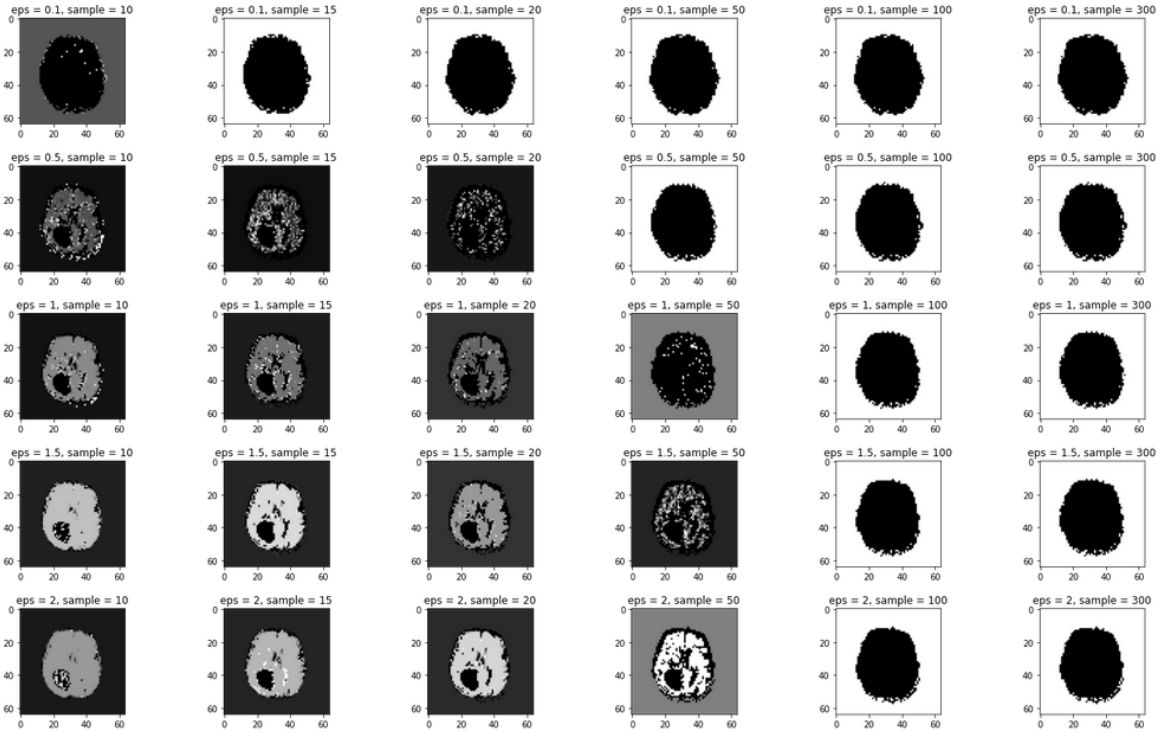


Agglomerative Clustering :

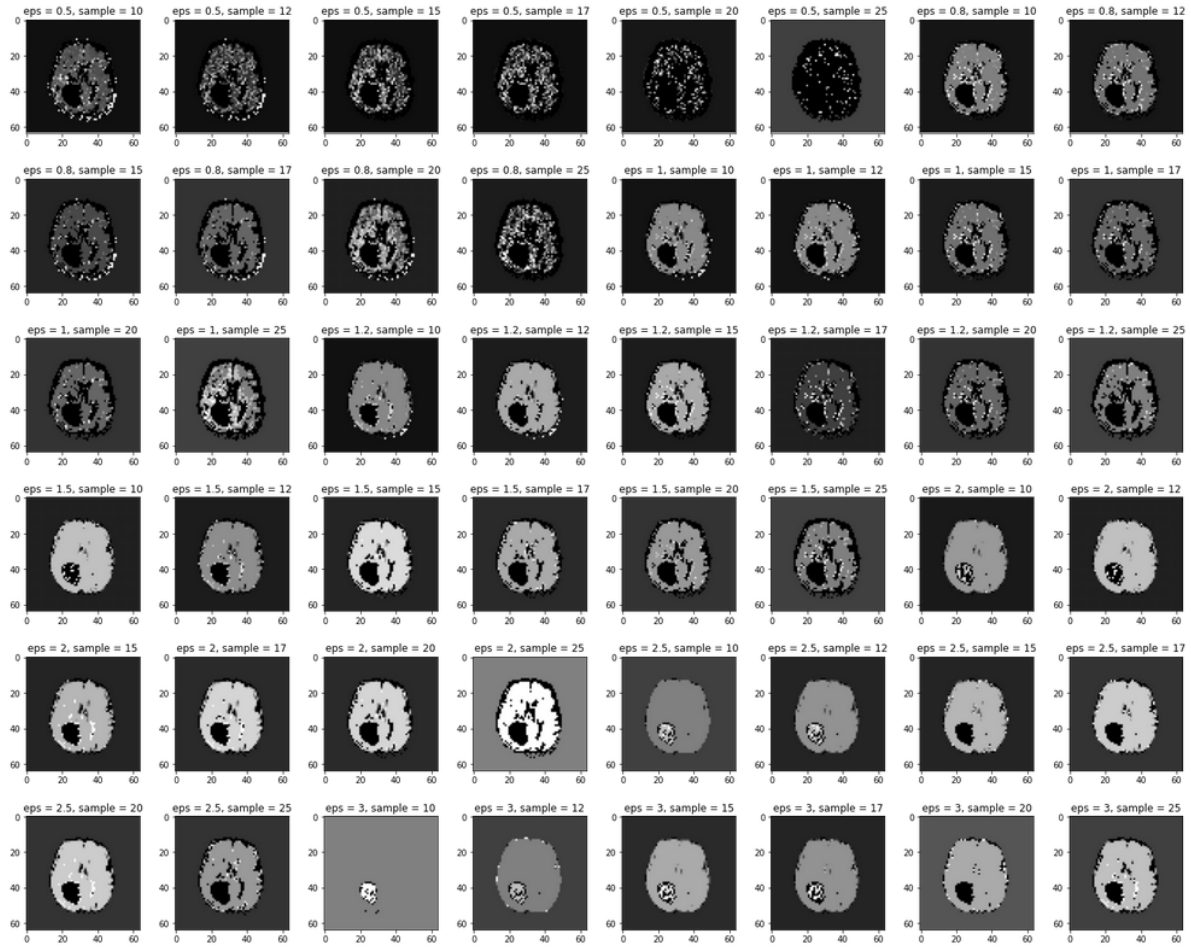


DBSCAN :

large tuning :

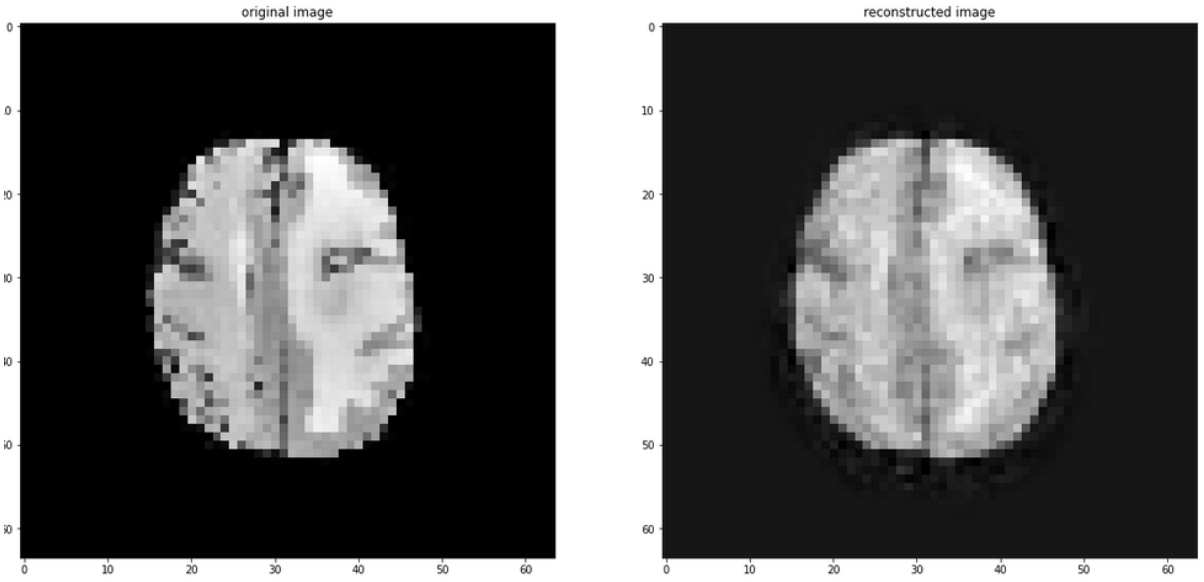


fine tuning:

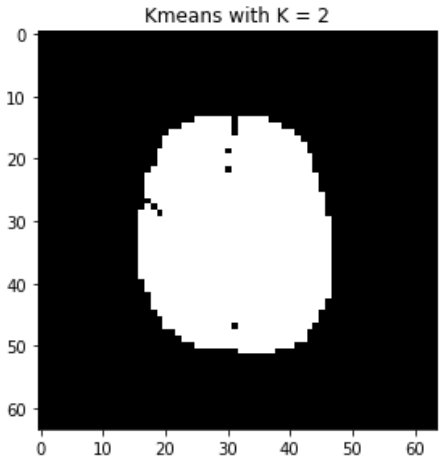


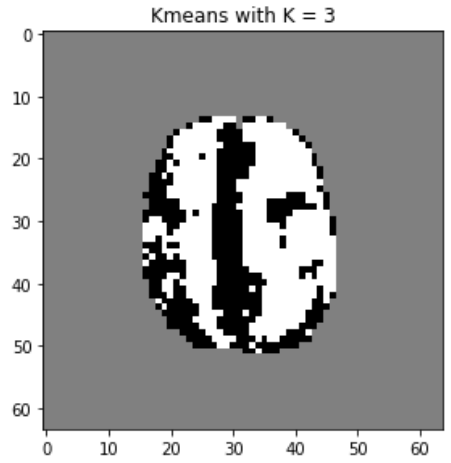
Annexe 2 : MRI image without tumor

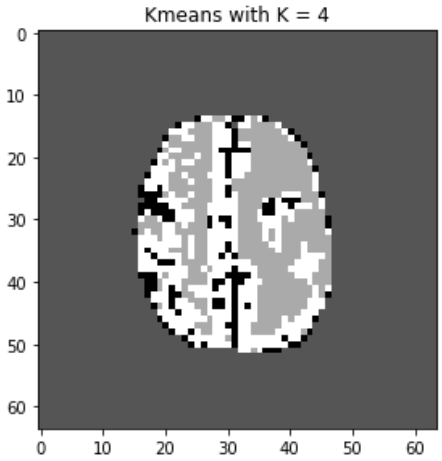
PCA reconstruction :

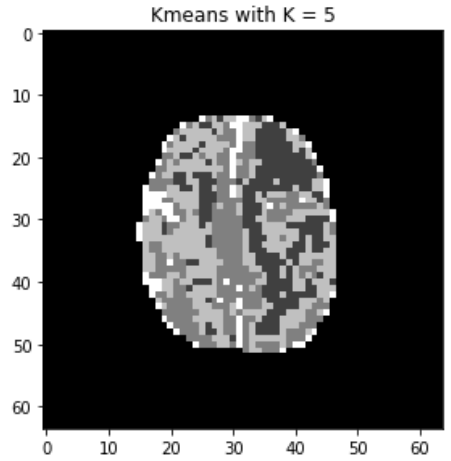


Kmeans :

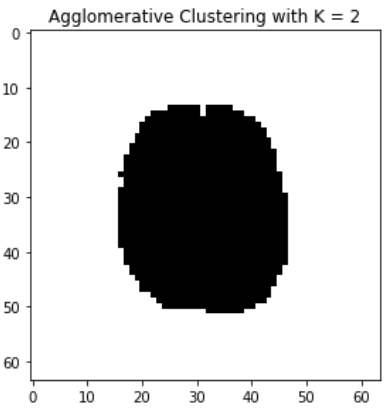


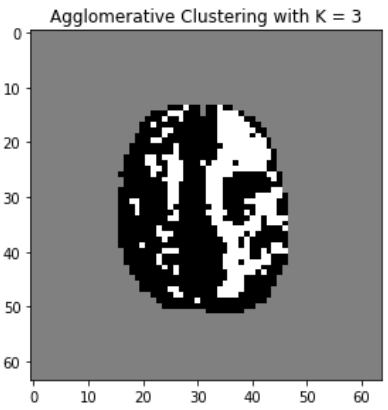


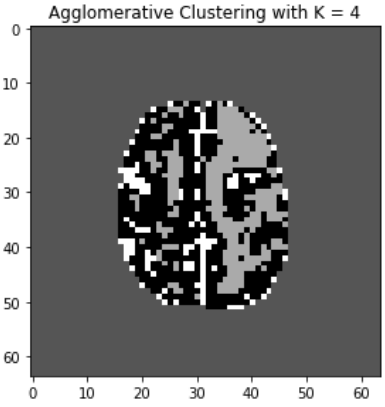


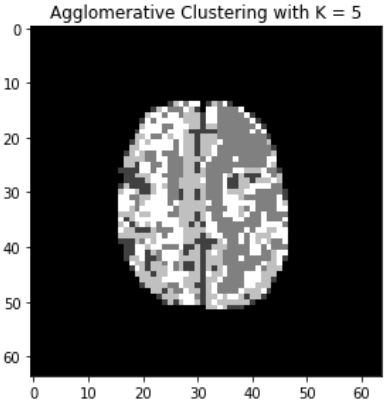


Agglomerative Clustering :



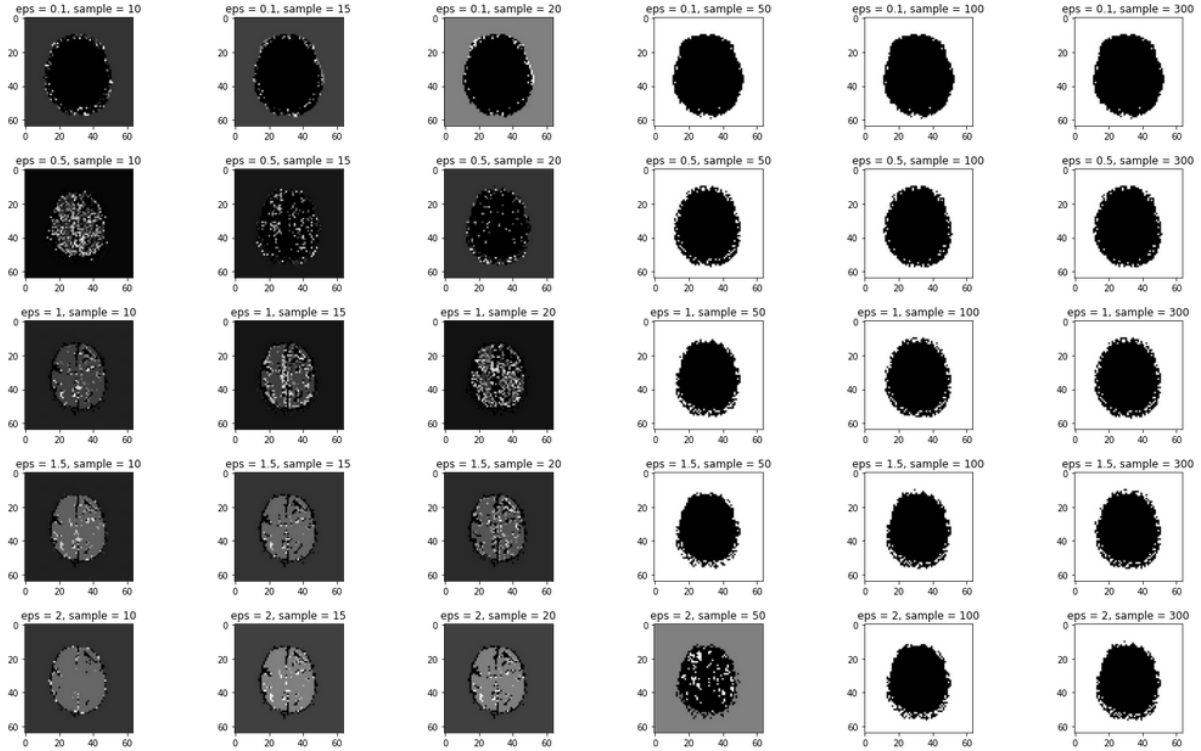




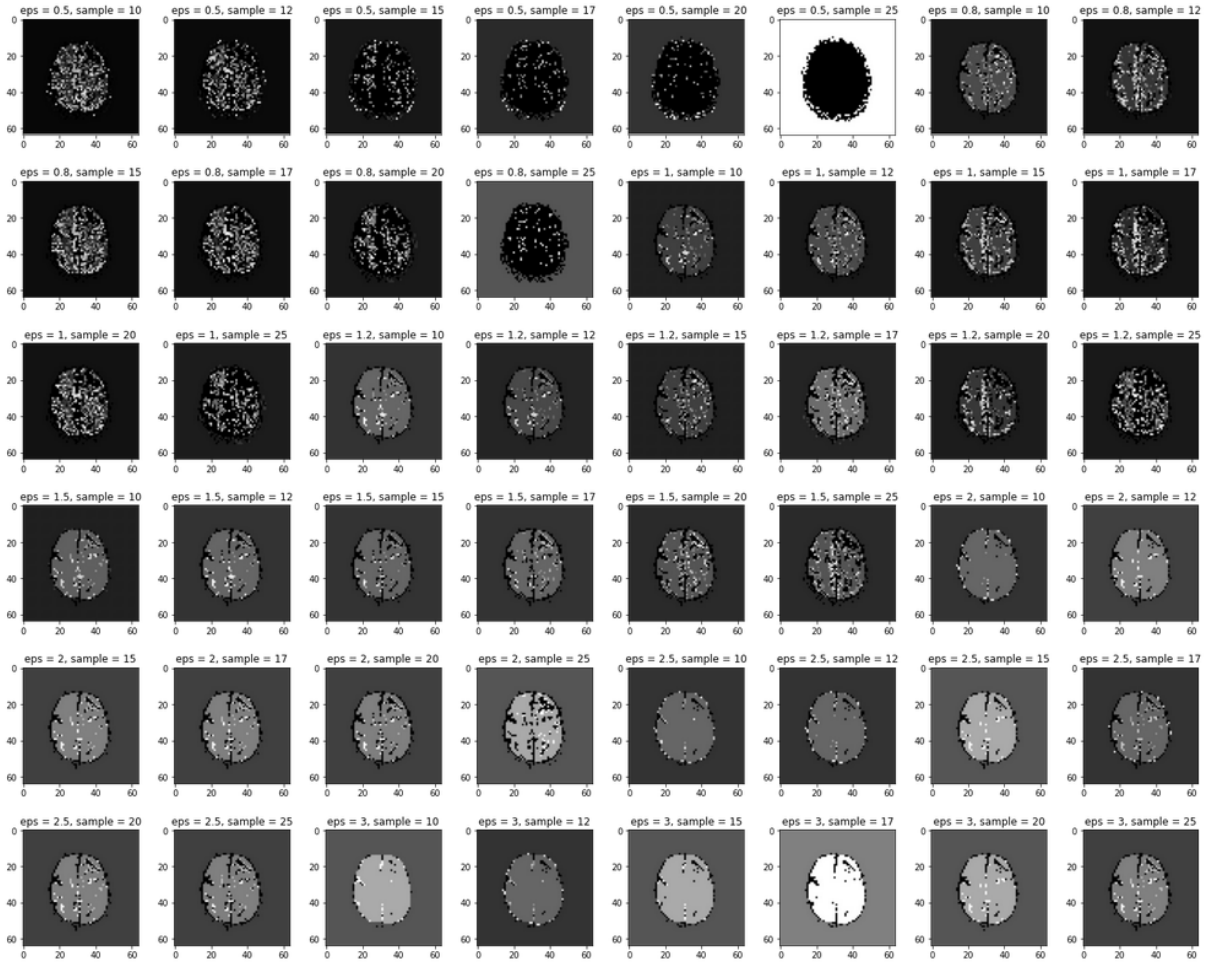


DBSCAN :

large :



fine :


eps = 0.5, sample = 10
eps = 0.5, sample = 12
eps = 0.5, sample = 15
eps = 0.5, sample = 17
eps = 0.5, sample = 20
eps = 0.5, sample = 25
eps = 0.8, sample = 10
eps = 0.8, sample = 12
eps = 0.8, sample = 15
eps = 0.8, sample = 17
eps = 0.8, sample = 20
eps = 0.8, sample = 25
eps = 1, sample = 10
eps = 1, sample = 12
eps = 1, sample = 15
eps = 1, sample = 17
eps = 1, sample = 20
eps = 1, sample = 25
eps = 1.2, sample = 10
eps = 1.2, sample = 12
eps = 1.2, sample = 15
eps = 1.2, sample = 17
eps = 1.2, sample = 20
eps = 1.2, sample = 25
eps = 1.5, sample = 10
eps = 1.5, sample = 12
eps = 1.5, sample = 15
eps = 1.5, sample = 17
eps = 1.5, sample = 20
eps = 1.5, sample = 25
eps = 2, sample = 10
eps = 2, sample = 12
eps = 2, sample = 15
eps = 2, sample = 17
eps = 2, sample = 20
eps = 2, sample = 25
eps = 2.5, sample = 10
eps = 2.5, sample = 12
eps = 2.5, sample = 15
eps = 2.5, sample = 17
eps = 2.5, sample = 20
eps = 2.5, sample = 25
eps = 3, sample = 10
eps = 3, sample = 12
eps = 3, sample = 15
eps = 3, sample = 17
eps = 3, sample = 20
eps = 3, sample = 25