**Lab Assignment #3: Hierarchical Clustering**

This assignment will be similar to Assignment 2 but we will use hierarchical clustering in place of K-Means.

**1. Data Preparation**

Since this assignment builds on the previous assignment, the preprocessing phase of the Olivetti dataset consists of a stratified split of the data samples. Furthermore, given the limited number of samples, we opted for reserving 80% of the set for training and the remainder equally split between validation and test sets.

A screen shot of a computer code

Description automatically generated

**2. The classification model**

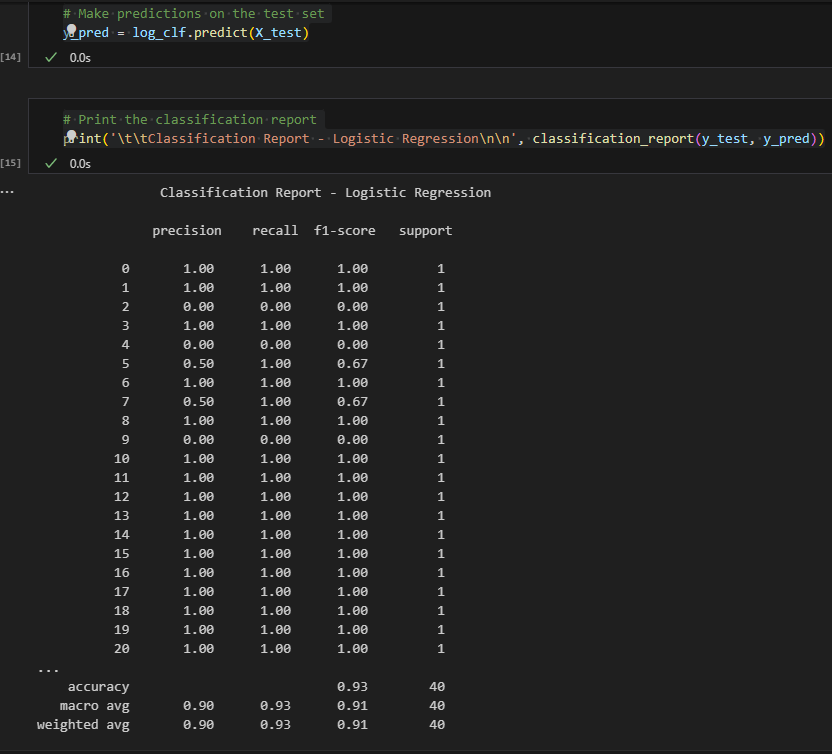
This turn, differently from the previous assignment, we opted to train a Logistic Classification model to test its ability to classify the Olivetti dataset’s images.

The screenshot below demonstrates that the model does an excellent job with the features in their original format, reaching a cross-validation score of roughly 94%.

A screenshot of a computer program

Description automatically generated

Additionally, the same classifier proved its generalization capacity by achieving an overall accuracy score of 93% in the test set,



We noticed that the model misclassified instances like labels 5 and 7. Therefore, we decided to plot label 7 to understand how tricky that target variable could be.

A collage of different facial expressions

Description automatically generated

**3. Clustering Analysis**

We then proceeded with the silhouette score analysis for Scikit Learn’s AgglomerativeClustering algorithm for the three requested metrics (Euclidean, Minkowski, Cosine).

The below Graph indicates how the silhouette score progressed in a range of 2 to 200 clusters for the three metrics. We can clearly see that the Cosine ‘affinity’ achieved much higher silhouette scores.

A graph of different sizes and colors

Description automatically generated

In possession of the number of clusters that return the highest silhouette score for each metric, we use ‘AgglomerativeClustering’ to compress/reduce the dataset by assigning data points to clusters and replacing each data point with their cluster centroid.

A screenshot of a computer program

Description automatically generated

With that step completed, we’ve split the data based on each compression, re-fit the classifier onto the data and performed new classifications as per the images below.

A computer screen shot of a program

Description automatically generated

A screen shot of a computer program

Description automatically generated

A screenshot of a computer program

Description automatically generated

From the cross-validation scores in the images above, we have a good indication that the compression using the Euclidean metric resulted in a higher score and, therefore, a lower loss of variability.

A screenshot of a computer

Description automatically generated

**4. K-Means Clustering for Dimensionality Reduction**

We subjected the training data to K-Means clustering across a range of potential cluster sizes, evaluating the silhouette score for each configuration.

A screenshot of a computer program

Description automatically generated

The silhouette analysis revealed the highest score for a configuration with 79 clusters.

A screen shot of a graph

Description automatically generated

However, reducing our data's dimensionality to these 79 clusters and reapplying our SVM classifier led to decreased accuracy. This suggests that the centroids of the 79 clusters might not capture the dataset's underlying variance adequately.

A screenshot of a computer

Description automatically generated

**5. DBSCAN Clustering**

Turning our attention to DBSCAN, we focused on optimizing its two primary hyperparameters: eps and min\_samples. Our goal was to approximate the natural division of the dataset, targeting 40 clusters, corresponding to the 40 distinct individuals.

A screenshot of a computer program

Description automatically generated

Unfortunately, the high dimensionality combined with the scarcity of samples made accurate clustering challenging. DBSCAN often misgrouped different individuals into the same cluster.

A collage of a person's face

Description automatically generated

However, in certain configurations, DBSCAN showed potential, delivering reasonably accurate clusters.

A collage of a person's face

Description automatically generated

**Conclusion**

This assignment showcased the challenges of clustering high-dimensional data with a limited number of samples. While SVM delivered decent classification accuracy on the original data, dimensionality reduction via K-Means and clustering via DBSCAN were less straightforward. Fine-tuning and further experimentation, possibly with other techniques or algorithms, could lead to better results in future endeavors.