3945 – Advanced Machine Learning

Home Assignment 1 – Unsupervised Learning

Abstract

In this assignment, we explored the different methodologies and concepts in unsupervised learning. Unsupervised learning is a type of machine learning that allows finding patterns in data without “ground truth”. This can be used for various tasks, such as
Clustering: the task of grouping data points together that are similar.
Dimensionality reduction: reducing the number of features in a dataset by extracting new features from the raw data.

Data

We used the MNIST dataset: a large database of handwritten digits. It is commonly used for training various image processing systems and other ML tasks. It contains 60,000 training images and 10,000 testing images. Each image is a 28x28 pixel grayscale image of a handwritten digit. The images are labeled with the corresponding digit.
Due to most of the algorithms used in our work being time-consuming, we **downsampled** the full MNIST datasets into a balanced training dataset sized **5000**, and a corresponding test dataset sized **1000** samples.

Part 1 - Clustering

We experimented with 3 different clustering methods (KMeans, Agglomerative, BatchKmeans) in the following manner:

1. We clustered the training set using raw features (ignoring the labels)
2. We calculated the clusters silhouette score.
3. We trained an MNIST classifier based on the cluster assignments, where a majority vote of the samples in that cluster assigns the cluster’s label.
4. We analyzed the classifier quality based on the train/test set.

Results

Reminder: $Silhouette\ Score\ =\ (b\ -\ a)/max(a, b)$ where:
a is the average intra-cluster distance and *b* is average inter-cluster distance. We got results very close to 0 (0.04-0.06), indicating that the clusters were not separable. We assume that possible reasons for that are:

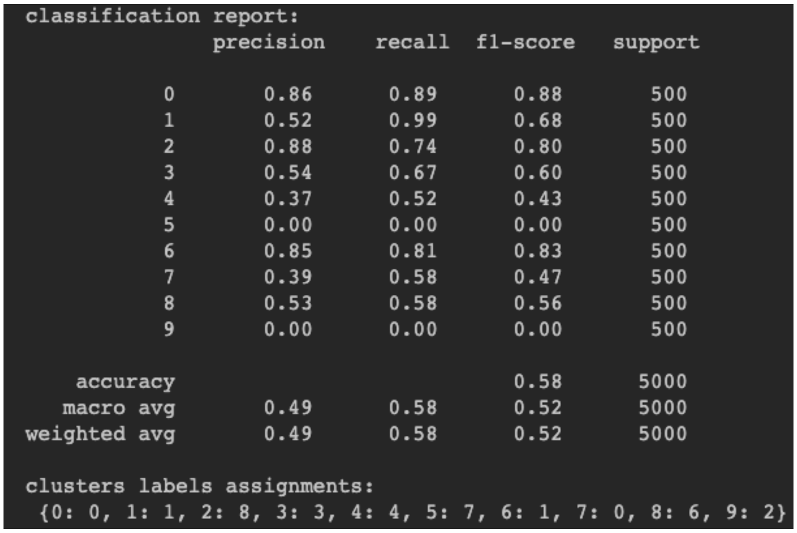
1. The data was not normalized (as required by the instructions of the assignment), and that could lead to issues with the robustness of the clustering algorithm **and** the actual

Silhouette coefficient calculation since outliers tend to decrease the results (large intra-cluster distance (*a*))

2. We used clustering methods that assign a cluster to ALL samples, including outliers (unlike DBSACN, where a sample can be assigned an “outlier” label). This can cause a degradation in the Silhouette score since “noise” is presented to the clusters and decreases the inter-cluster distance(*b*)

We suggest running the same experiment using a normalized dataset to re-evaluate the clustering quality of the proposed methods.

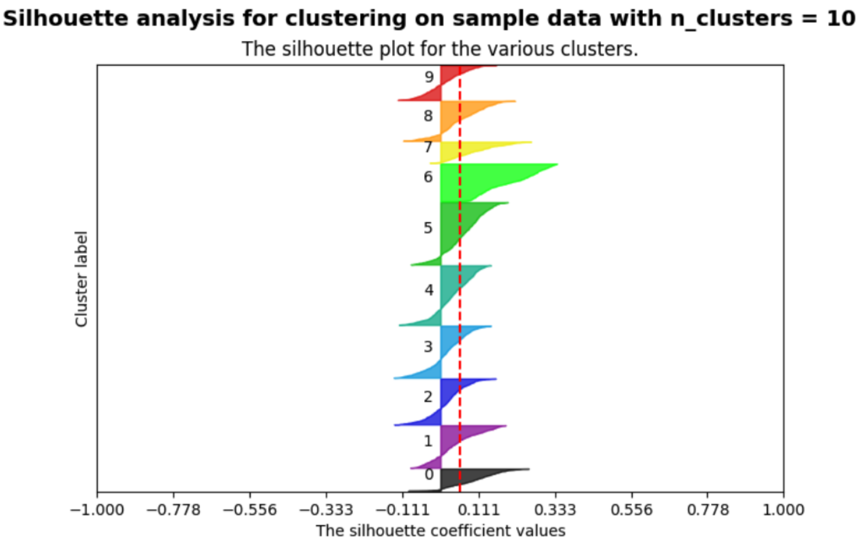
We got better results in classification based on cluster assignments using an Agglomerative clustering-based classifier. We assume this happens because the clustering is built using the **ward** linkage method (the method we used in our work) that minimizes the variance of the clusters during the process and results in better separable clusters. Generally, we saw better classification results for classes that were better clustered than others. I.e. we see in the following case for **KMneas** clustering and class `**1**`:



```
classification report:
              precision    recall  f1-score   support

           0       0.86      0.89      0.88       500
           1       0.52      0.99      0.68       500
           2       0.88      0.74      0.80       500
           3       0.54      0.67      0.60       500
           4       0.37      0.52      0.43       500
           5       0.00      0.00      0.00       500
           6       0.85      0.81      0.83       500
           7       0.39      0.58      0.47       500
           8       0.53      0.58      0.56       500
           9       0.00      0.00      0.00       500

    accuracy                           0.58      5000
   macro avg       0.49      0.58      0.52      5000
weighted avg       0.49      0.58      0.52      5000

clusters labels assignments:
 {0: 0, 1: 1, 2: 8, 3: 3, 4: 4, 5: 7, 6: 1, 7: 0, 8: 6, 9: 2}
```

Classification results for KMeans-based classifier

We can see that `1` has a high recall value (most samples labeled `1` were predicted correctly) but low precision (many non-labeled `1`s were also predicted 1). There is a good explanation for that when looking at the silhouette plot of the samples:



We can see that clusters 1 and 6 were assigned the label `1`, but cluster 1 has a low silhouette score and is probably responsible for the degradation in the overall class `1` precision.

Part 2 - Dimensionality Reduction

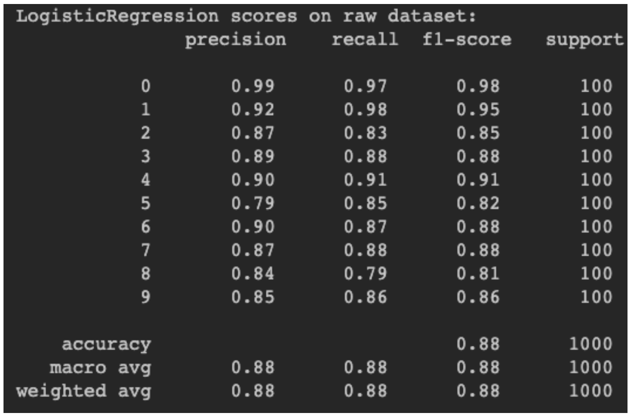
We used 3 methods for feature extraction and reduced the dataset dimensions (PCA, T-SNE, and UMAP) in the following manner:

1. We performed dimensionality reduction to two features.
2. We plotted the new dataset features.
3. We trained an MNIST logistic regression classifier based on the new features.

We analyzed the classifier quality based on the train/test set.
Overall, UMAP achieved the best results regarding visual separability in a plot and classifier scores on the train/test sets.
Next was TSNE, and lastly arrived PCA, with the worst classification score and visual separability. Manifold-based dimensionality reduction methods work better on the MNIST dataset than PCA because they can capture the non-linear structure of the data. PCA is a linear dimensionality reduction method that can only capture linear relationships between the data points. However, the MNIST dataset is non-linear, meaning the data points are not linearly related. This is because the handwritten digits are not perfectly aligned, and there is some variation in how they are written. This is why UMAP and T-SNE performed better in our case.

Part 3: Raw data classification

In the last part, we trained a logistic regression classifier using the entire raw dataset. We used the grid-search CV method to perform hyperparameter tuning and achieved the following results, using 'C': 0.001, 'penalty': 'l2' parameters.



```
LogisticRegression scores on raw dataset:
              precision    recall  f1-score   support

           0       0.99      0.97      0.98       100
           1       0.92      0.98      0.95       100
           2       0.87      0.83      0.85       100
           3       0.89      0.88      0.88       100
           4       0.90      0.91      0.91       100
           5       0.79      0.85      0.82       100
           6       0.90      0.87      0.88       100
           7       0.87      0.88      0.88       100
           8       0.84      0.79      0.81       100
           9       0.85      0.86      0.86       100

    accuracy                           0.88      1000
   macro avg       0.88      0.88      0.88      1000
weighted avg       0.88      0.88      0.88      1000
```