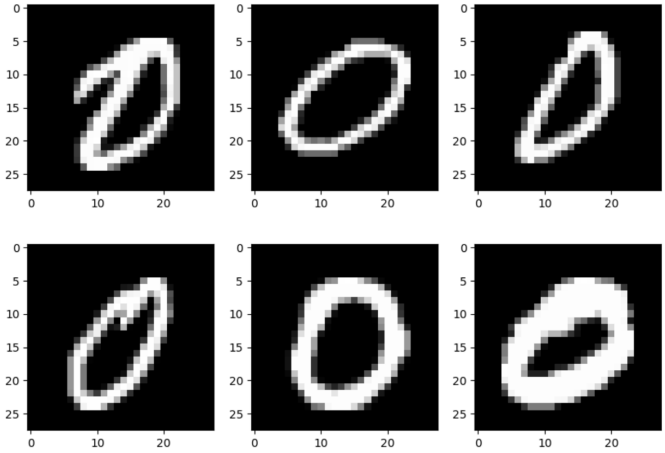
Due: April 30th, 2023
CS3946: Advanced Machine Learning

Home Assignment 1: Unsupervised Learning

In this assignment, we explored unsupervised learning techniques for clustering and dimensionality reduction on the MNIST database of handwritten digits. In the data exploration phase, we loaded the dataset and explored the different styles of digits:



Part 1: Clustering

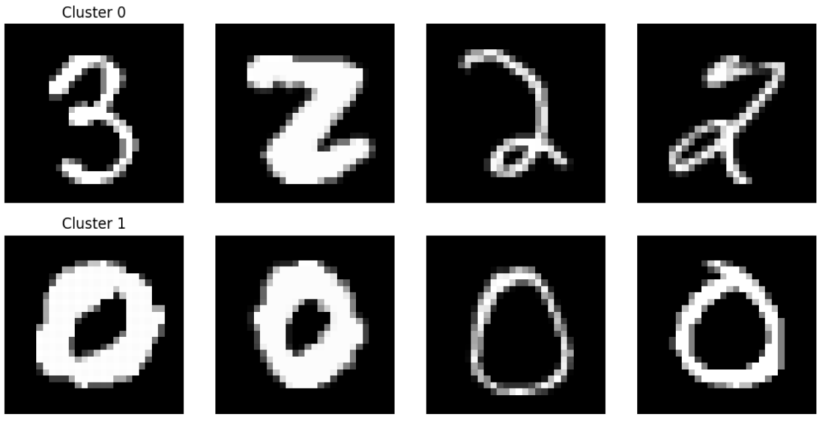
In the first part of the assignment, we chose four different clustering methods and for each: clustered the data using the raw pixel level values, evaluated the clustering result using standard evaluation metrics, visualized the clusters, built a simple classifier based on the clustering results, evaluated the performance of the classifier and explored different parameters for the clustering to evaluate their effects on the results. We evaluated the clustering methods using the silhouette score (measures how well separated clusters are from -1 to 1), homogeneity score (measures how much the clusters contain only samples from a single class from 0 to 1), completeness score (measures how much all members of a given class are assigned to the same cluster from 0 to 1) and the V-measure score (combines the homogeneity and complete score to provide an overall evaluation of clustering performance from 0 to 1). We used Logistic Regression as the classifier for each of the methods and evaluated the classifier using accuracy (measures the percentage of correct predictions made by the classifier out of the total number of samples from), precision (measures the percentage of true positive predictions amount all the position predictions made by the classifier from) and recall (measures the percentage of true positive predictions among all the actual positive samples from). We initiated the clustering algorithms with 4 clusters and only selected classes 0, 1, 2, 3 (under sampling) to optimize the processing speed.

Method 1: Gaussian Mixture Model (GMM)

A GMM is a probabilistic model that represents a dataset as a mixture of several Gaussian distributions, where each distribution in the mixture represents a cluster in the data. The classifier uses this model to assign labels to new data points based on the probability of the point belonging to each cluster. It received the following evaluation results:

- Silhouette score of 0.075: This is close to 0 so it indicates that the clusters may overlap and that the clusters are not well separated.
- Homogeneity score of 0.54: Indicates that the clusters are moderately homogenous but the clusters may contain samples from more than one class
- Completeness score of 0.56: Indicates that a majority of the samples are assigned to the same cluster, but there are still samples assigned to different clusters
- V-Measure score of 0.55: Indicates that the GMM performs okay but definitely has room for improvement

This performance can be visualized for better insight and shows that there is a clustering error:



The Logistic Regression classifier received the following evaluation results:

- Accuracy Score 80.9% : the classifier correctly classified most of the samples
- Precision Score 82%: a large majority of the positive samples were correctly identified
- Recall Score 80.8%: the classifier is able to correctly identify most positive samples

We explored different parameters by initializing the covariance type to be diagonal (so that each component has its own diagonal covariance matrix) and for the parameter initialization method to randomly select data points. However, the cluster performance decreased in comparison.

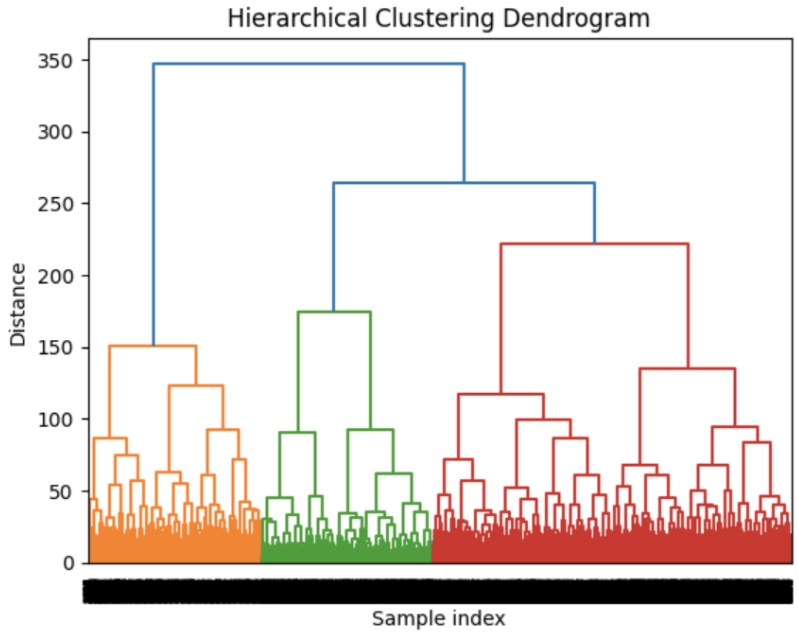
Method 2: Agglomerative Clustering

Agglomerative Clustering is a hierarchical clustering algorithm that works by iteratively merging the closest pairs of data points until all points belong to a cluster. It received:

- Silhouette score of 0.109: Indicates that there is an overlap between the clusters
- Homogeneity score of 0.904: each cluster mostly contains data points from a single class
- Completeness score of 0.904: more data points from the same class are assigned to the same cluster
- V-Measure score of 0.904: combines homogeneity with completeness to suggest most of the data points were evaluated well, with some exceptions

It is interesting to note that the silhouette score is quite poor compared to the overall v-measure score. This shows that even though it is not effectively separating the points to distinct clusters, the actual resulting clusters contain mostly data points from a single class.

The performance can be visualized using a dendrogram, where each leaf represents a data point and each node represents a cluster:



The Logistic Regression classifier received the following evaluation results:

- Accuracy Score 96.5% : almost all of the samples were classified correctly
- Precision Score 96.4%: the proportion of false positives among positive was very low
- Recall Score 96.5%: low number of false negatives

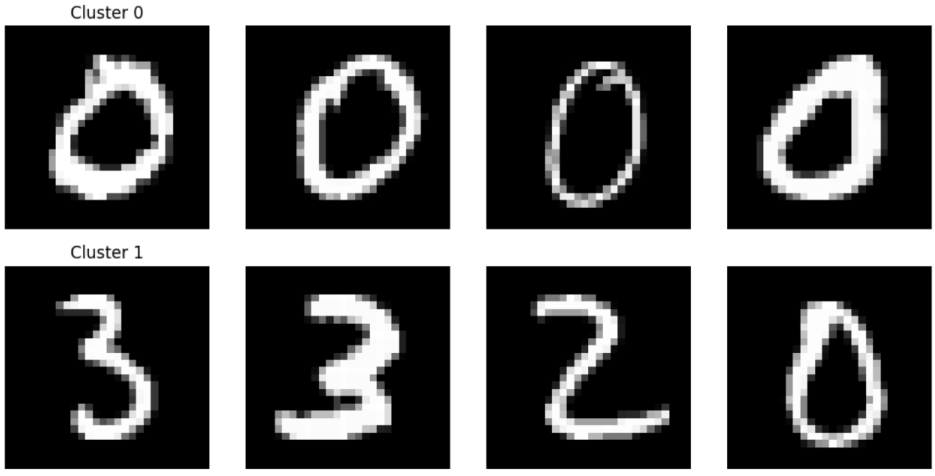
We explored different parameters by initializing the linkage criterion to complete and the metric to Manhattan. However, the cluster performance decreased in comparison.

Method 3: K-Means

K-Means is an algorithm used for grouping unlabeled data into clusters based on their similarity, and it works by assigning each data point to one of K clusters. It aims to minimize the sum of squared distances between each data point and its assigned centroid but it does not guarantee that it will find the global optimal clustering. It received the following evaluation results:

- Silhouette score of 0.120: Indicates that there is an overlap between the clusters
- Homogeneity score of 0.662: Indicates that the clusters are moderately homogenous but the clusters may contain samples from more than one class
- Completeness score of 0.666: a majority of the samples are assigned to the same cluster, but there are still samples assigned to different clusters
- V-Measure score of 0.664: performs moderately well but definitely has room for improvement

This performance can be visualized for better insight and shows that there is a clustering error:



The Logistic Regression classifier received the following evaluation results:

- Accuracy Score 87.3% : most of the samples were classified correctly
- Precision Score 87.9%: proportion of false positives among positive was moderately low
- Recall Score 87.1%: relatively number of false negatives

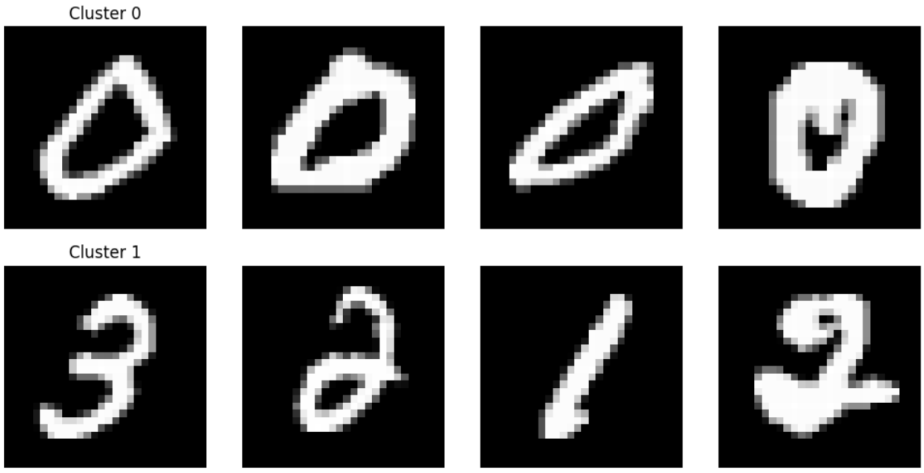
We explored different parameters by initializing the algorithm with the Lloyd algorithm . However, the cluster performance was not affected.

Method 4: Mini Batch K-Means

Similar to K-Means, Mini Batch randomly samples a small batch of data points from the entire dataset at each iteration (instead of using the entire dataset). We used a mini batch size of 256. It received the following evaluation results that are quite similar to K-Means:

- Silhouette score of 0.105
- Homogeneity score of 0.554
- Completeness score of 0.556
- V-Measure score of 0.555

This performance can be visualized for better insight and shows that there is a clustering error:



The Logistic Regression classifier received the following evaluation results, that are very similar but slightly less good than classic K-Means:

- Accuracy Score 81.3%
- Precision Score 82.1%
- Recall Score 81.2%

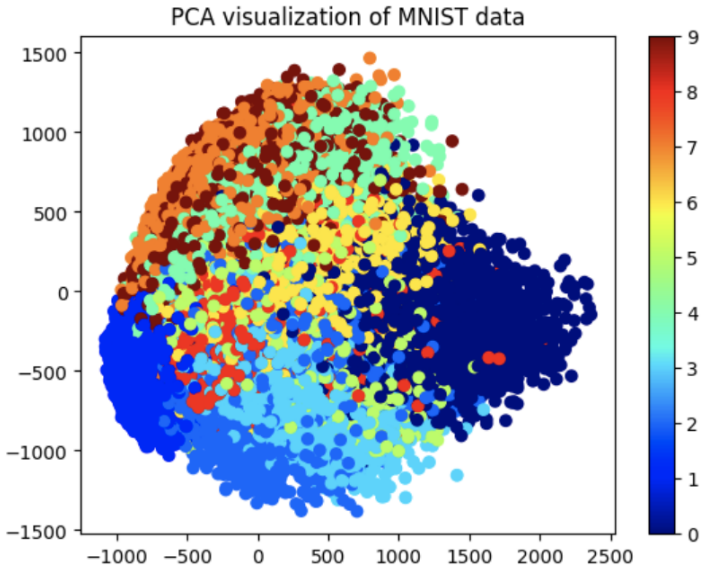
We explored different parameters by initializing the centroids randomly. This improved the v-measure score, completeness score, homogeneity score and silhouette score of the performance.

Part 2: Dimensionality Reduction

In the second part of the assignment, we choose three different dimensionality reduction methods and for each: reduced the dimensionality of the data, visualized the first few features in the new space, trained two classifiers on the data in the new space, evaluated the performance of the classifiers and explored different parameters for the dimensionality reduction / evaluated their effect on the results. To evaluate the performance of the classifiers, we used the built in score() function from the scikit-learn library which is a utility function for measuring accuracy. For some clustering methods, we only selected classes 0, 1, 2, 3 to optimize the processing speed and 2 as the number of components. We trained Logistic Regression and KNN as classification models to evaluate the dimensionality reduction results.

Method 1: Principal Component Analysis (PCA)

PCA reduces the dimensionality while retaining as much of the original variation in the data as possible by identifying the directions in which the data varies the most and represents the data in a new coordinate system that is defined by these directions. So the first principal component (PC) is the direction that captures the most variation in the data, etc. Below is a visualization of the reduced dimensions, which shows that the clusters are overlapping so perhaps the PCs did not capture all the important variation in the data and there exists some outliers:

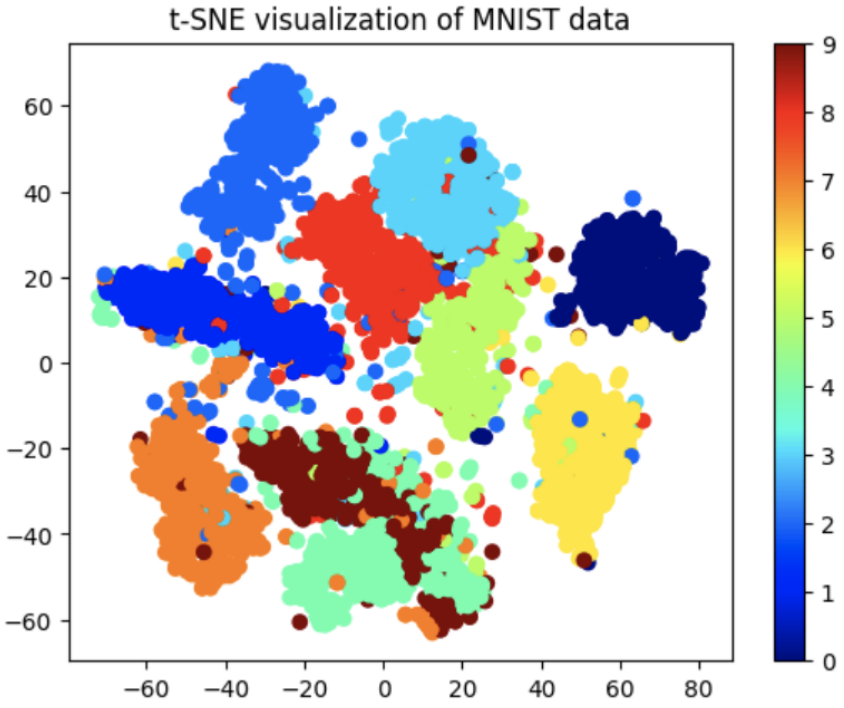


Both Logistic Regression and KNN scored about 44% for the accuracy of the model, which suggests that the model is not performing so well and required optimization. This makes sense due to the fact that the PC clusters were quite overlapping.

We explored different parameters by setting data whitening to True. This Improved the LogReg classifier only slightly. It did not have an effect on KNN.

Method 3: t-Distributed Stochastic Neighbor Embedding (t-SNE)

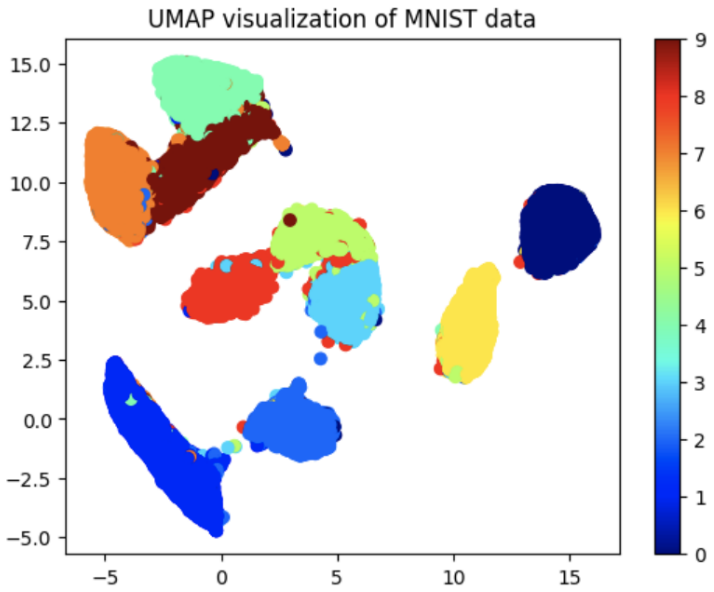
t-SNE uses a probabilistic approach to map each high dimensional data point to a lower dimensional point, while preserving the relationships and similarities between the data points as much as possible. We used undersampling to improve runtime, but this resulted in decreased accuracy in the results. Below is a visualization of the reduced dimensions, which shows that the clusters are better separated but there is outlying data points within other clusters:



Both Logistic Regression and KNN scored quite poorly (13% and 20%, respectively) for the accuracy of the model, due to the fact that the data was undersampled to improve runtime. It is difficult to draw complete conclusions from these numbers, especially given the visualization where the dimensions are relatively well separated. One would expect the score to be at least greater than that given to PCA, due to the fact that the dimensions are better clustered.

Method 3: Uniform Manifold Approximation and Projection (UMAP)

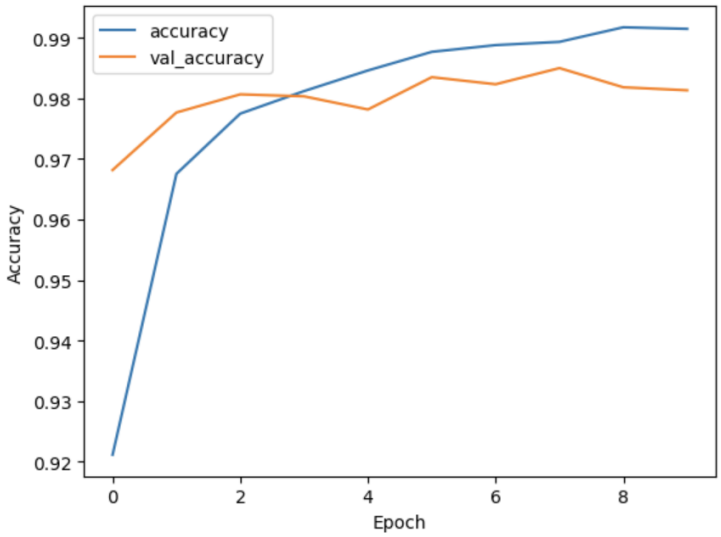
UMAP is based on the idea of constructing a weighted graph representation of the data, where each data point is connected to its nearest neighbors. The graph works to minimize the distance between connected points in the graph. Below is a visualization of the reduced dimensions, which shows the clusters are quite well separated overall (with the exception of a few groups of clusters) and the outliers fairly minimized:



Both Logistic Regression and KNN worked quite well, scoring more than 90% for accuracy. This makes sense given the relatively clean degree of separation in the graph.

Part 3: Classification of the raw data

In the third part of the assignment, we trained a classification algorithm on the raw pixel values, explored different parameters for the classification model and evaluated the results of the model. We used a Multi-Layer Perceptron (MLP) model with two hidden layers, each hidden layer with 512 units, a ReLu activation and a dropout layer after each hidden layer (with 20% dropout rate) to prevent overfitting. The output layer has 10 units and uses softmax activation to output the class probabilities. We compiled the model with the Adam optimizer and categorical cross entropy loss. The model was trained with 10 epochs with a batch size of 128 and a 10% validation split. After training the model, we plotted the training history to visualize the training and validation accuracy over the epochs.



We evaluated the model on the test set by computing the accuracy, classification (including precision, recall and F1-score) and the confusion matrix. Lastly, we plotted some of the misclassified images with their true and predicted labels. It received an accuracy score of over 97%, which is quite good.

Classification report:

	precision	recall	f1-score	support
0	0.98	0.99	0.99	980
1	0.99	0.98	0.99	1135
2	0.98	0.98	0.98	1032
3	0.99	0.97	0.98	1010
4	0.99	0.98	0.99	982
5	0.96	0.98	0.97	892
6	0.96	0.99	0.97	958
7	0.98	0.98	0.98	1028
8	0.98	0.95	0.97	974
9	0.97	0.99	0.98	1009
accuracy			0.98	10000
macro avg	0.98	0.98	0.98	10000
weighted avg	0.98	0.98	0.98	10000

To explore different parameters of the classification model, we reduced the number of neurons in the hidden layer from 512 to 256 in the first hidden layer and 128 in the second

hidden layer. We also increased the dropout rate to 50% to increase the regularization effects of the model and decrease batch size to 64 while increasing the number of epochs to 20 to allow the model to train for longer. This increased the performance accuracy to over 98%.

Part 4: Summary of findings

Here is a table that summarizes the results obtained in part 1: clustering:

	Silhouette	V-Measure	Accuracy	Precision	Recall
GMM	0.075	0.55	80.9%	82%	80.8%
Agglomerative	0.109	0.904	96.5%	96.4%	96.5%
K-Means	0.120	0.664	87.3%	87.9%	87.1%
Mini Batch K-Means	0.105	0.555	81.3%	82.1%	81.2%

While Agglomerative clustering performed the best in terms of homogeneity and completeness, K-Means performed best in terms of the silhouette score. This is probably due to the fact that K-Means seeks to minimize the sum of squared distances so it tends to produce clusters that are tightly packed around their centroids (which can result in higher silhouette scores). Agglomerative clustering recursively merges clusters based on their distance so it is influenced by outlier points that don't belong to any cluster (which can result in lower silhouette scores). For the classification after the clustering, Agglomerative performed the best across the board.

Here is a table that summarizes the results obtained in part 2: dimensionality reduction:

	Classifier Scores
PCA	44% accuracy
t-SNE	20% accuracy
UMAP	90% accuracy

It should be noted that t-SNE was undersampled in order to optimize runtime and therefore the accuracy score is not a fair comparison to the others. PCA had the most overlapping dimensions when visualized, while UMAP had the best separation. Therefore, it makes sense for UMAP to have the best classification accuracy score.