3945_HW1_011996279_207576463_report

April 28, 2023

0.1 3945_HW1 Summary Report

- student_ID_1 = 011996279
- student_ID_2 = 207576463

0.1.1 Part 4: Summary

6) Summarize the results obtained in parts 1-3 compare the results of different methods and suggest possible explanations.

Summary of Methods: In summary, we have performed clustering, dimensionality reduction, and classification on the MNIST dataset, and we took several different approaches to each task, and tested several combinations of parameters for each approach, evaluating each according to its resulting classification accuracy, along with other scoring metrics for the relevant tasks.

For **clustering**, we chose to use **K-Means, DBSCAN, Agglomerative Clustering, and Birch** as clustering algorithms to test. We evaluated their performance using silhouette score, adjusted rand index (ARI), completeness score, and homogeneity score. We then built a simple k-nearest neighbors (KNN) classifier using the cluster labels and evaluated its performance in terms of accuracy.

For **dimensionality reduction**, we used the **PCA, t-SNE, and UMAP** algorithms, and then also added **ICA and TruncatedSVD** for kicks. We visualized the first few features of reduced data in the new space and trained a logistic regression classifier on it. We evaluated the classifier's performance in terms of accuracy, plotted against the various hyperparameter values for that algorithm along the x-axis. We also looked at the component-wise and cumulative explained variance ratio in the new reduced space for PCA and TruncatedSVD.

For the stand-alone **classification** task, we trained **logistic regression (LoR) and Random Forest classifiers** on the raw pixel values of the dataset and evaluated its performance using a classification report, which includes precision, recall, f1 score, and of course accuracy, and also looked at a confusion matrix and error rates to visually inspect the performance of the classifier across the different digits/classes and to analyze error attribution. We also experimented briefly with a gradient boost classifier and a multi-layer perceptron (MLP) classifier for the sake of completeness.

0.1.2 Summary of Results - High-Level Algorithm Comparisons:

Clustering We applied KMeans, DBSCAN, Agglomerative clustering, and Birch algorithms on the dataset and evaluated their performance using Silhouette Score, Adjusted Rand Index (ARI), Completeness, Homogeneity, Classifier Accuracy, and Purity metrics, and present those comparisons below.

- KMeans Clustering:
  - Silhouette: 0.05
  - ARI: 0.36
  - Completeness: 0.49
  - Homogeneity: 0.49
  - Accuracy: 0.60
  - Purity: 0.60

KMeans, a centroid-based clustering algorithm, achieved moderate performance. Its low silhouette score indicates that the clusters may be overlapping, which is expected since the MNIST dataset contains images with similar features. The relatively higher accuracy and purity scores indicate that KMeans is able to create clusters that represent the digit classes reasonably well.

- DBSCAN Clustering:
  - Silhouette: -0.12
  - ARI: 0.13
  - Completeness: 0.61
  - Homogeneity: 0.35
  - Accuracy: 0.43
  - Purity: 0.30

DBSCAN, a density-based clustering algorithm, had poor performance. The negative silhouette score suggests that the clusters formed are not well-separated. This is likely because DBSCAN is sensitive to the choice of hyperparameters (e.g., eps and min_samples), and tuning these parameters is crucial for the algorithm to work effectively on the dataset.

- Agglomerative Clustering:
  - Silhouette: 0.03
  - ARI: 0.48
  - Completeness: 0.68
  - Homogeneity: 0.65
  - Accuracy: 0.67
  - Purity: 0.36

Agglomerative clustering, a hierarchical clustering algorithm, outperformed KMeans and DBSCAN in terms of ARI and homogeneity. The low silhouette score indicates some overlap between clusters. However, its higher ARI and homogeneity suggest that the algorithm can capture the underlying structure of the data more effectively.

- Birch Clustering:
  - Silhouette: 0.03
  - ARI: 0.56
  - Completeness: 0.71
  - Homogeneity: 0.68
  - Accuracy: 0.71
  - Purity: 0.71

Birch is an efficient, hierarchical, centroid-based clustering algorithm that works well on large datasets.

Dimensionality Reduction We applied PCA, t-SNE, UMAP, ICA, and Truncated SVD dimensionality reduction techniques and trained logistic regression classifiers on the reduced data. The best classifier accuracy for each method was:

- PCA: 90.9%
- t-SNE: 87.1%
- UMAP: 93.1%
- ICA: 82.5%
- Truncated SVD: 91.4%

UMAP and Truncated SVD outperformed other dimensionality reduction methods, indicating that they captured the intrinsic structure of the data more effectively, leading to better classification performance. PCA also performed reasonably well, but ICA and t-SNE had lower accuracies, possibly due to their inability to effectively separate the digit classes in the reduced feature space.

Stand-alone Classifiers The accuracy of the stand-alone classifiers was:

- Logistic Regression: 93%
- Random Forest: 97%

0.1.3 Summary of Results - Parameter Tuning:

- Clustering algorithms:
  - K-Means: For K-Means parameters, we tested the lloyd vs. elkan algorithms and k-means++ vs. random initiations, and the results across clustering metrics and classifier accuracy were entirely unaffected.
  - DBSCAN: For DBSCAN, we initially tested combinations of four different parameters, but ultimately had challenges with the l1/manhattan and l2/euclidean distance metrics (so defaulted to cosine pairwise distances) and found no difference between auto and brute algorithms, and therefore tested 6 combinations of min_samples and epsilon (eps) parameters. 0.125 was clearly the best epsilon in terms of classifier accuracy, but interestingly the other clustering metrics moved in unpredictable and opposing directions, so the choice of hyperparameters would really depend on the underlying optimization objective.
  - Agglomerative: For Agglomerative clustering we tested various values for linkage and distance metric, though we ultimately cut the linkage params down to only ward and complete, as average and single produced far inferior outcomes and also took forever to run. In the case of Agglomerative clustering, the ward linkage and euclidean distance param pairing was a clear standout across all evaluation metrics.
  - Birch: For Birch clustering, we tested the threshold and branching_factor parameters, which stipulate the subcluster splitting parameters for as new samples are added. Like K-Means, there was very little sensitivity to these parameters, with the exception of one combination - (0.7,60) - which led to meaningful improvements in every metric, with the exception of a slightly lower Silhouette score.

- Dimensionality reduction algorithms:
  - n_components - PCA, UMAP, ICA, and Truncated SVD: For these four algorithms, we tested various values for n_components in the resulting reduced space, and generally the resulting classifier accuracy was higher as the number of components increased, with

the exception of UMAP having a point of diminishing returns.
  - perplexity - t-SNE: for t-SNE, we tested several values of perplexity, and the classifier accuracy actually decreased for higher values of perplexity.
  - explained_variance_ratio - PCA and TruncatedSVD: For these two dimensionality reduction algorithms, the scikit-learn packages came with an "explained_var_ratio" attribute, which showed the ratio of the original data's variance explained by component in the new reduced space (ordered from highest to lowest as in PCA). In both cases, the first 10 components explained roughly half of the total variance.
- Stand-alone classifiers:
  - LoR: for the Logistic Regression classifier, we tested various values for C (an inverse regularization parameter) and tolerance for stopping criteria. Of these two, a lower value for C, which indicates a greater degree of regularization, resulted in higher classifier accuracy.
  - Random Forest: for the Random Forest classifier, we tested criterion (gini, entropy, log_loss), max_depth, and min_samples_split. Of these 3 parameters, the criterion had a negligible effect on accuracy, a larger threshold for min_samples_split had a marginally negative effect, and the max_depth was very material, as allowing greater max tree depth resulted in greater accuracy, as would be expected.

0.1.4 Summary of Results - Discussion:

Comparing the results, we observe that dimensionality reduction with PCA or UMAP followed by a logistic regression classifier yields better accuracy than using raw pixels directly. Clustering methods like KMeans and AgglomerativeClustering, when combined with a simple k-nearest neighbors classifier, may not perform as well as dimensionality reduction methods. This may be due to the fact that the clustering methods might not capture the intrinsic structure of the data as effectively as the dimensionality reduction methods.

The differences in the results of the different methods can be explained by the varying ability of each method to capture the underlying patterns in the data. Dimensionality reduction methods like PCA and UMAP can capture the intrinsic structure of the data more effectively, leading to better classification performance. Clustering methods, on the other hand, may not be as efficient in capturing the complex relationships between the features, leading to suboptimal classification performance.

Clustering: KMeans, a centroid-based clustering algorithm, displayed moderate performance on the dataset. The low silhouette score (0.05) suggests some overlap between clusters, which can be attributed to the inherent similarity between digit classes. Nonetheless, KMeans achieved relatively higher accuracy (0.60) and purity (0.60), signifying its ability to create clusters that represent digit classes reasonably well.

DBSCAN's poor performance can be explained by the nature of the algorithm as a density-based clustering method. The negative silhouette score (-0.12) indicates a lack of well-separated clusters, which can be attributed to the sensitivity of DBSCAN to its hyperparameters, eps and min_samples. Tuning these parameters is critical for DBSCAN to work effectively on the dataset. Additionally, the varying density of digit classes in the feature space likely posed challenges for the DBSCAN algorithm.

Agglomerative clustering, a hierarchical method, demonstrated better performance in terms of

ARI (0.48) and homogeneity (0.65) compared to KMeans and DBSCAN. While the low silhouette score (0.03) implies some overlap between clusters, Agglomerative clustering's ability to capture the underlying structure of the data more effectively may be explained by its bottom-up approach, which merges similar data points into progressively larger clusters.

Regarding the clustering algorithms, it is essential to consider the specifics of the MNIST dataset. The data contains handwritten digits with varying writing styles, leading to significant intra-class variations and inter-class similarities. This makes it challenging for clustering algorithms to create well-separated clusters. For example, digits like '4' and '9' can be easily confused due to their similar shapes. KMeans and Agglomerative clustering performed better in this context because they are less sensitive to the choice of hyperparameters and can better handle the inherent complexities of the data.

Comparing Birch to K-Means, DBSCAN, and Agglomerative clustering, we observe that Birch has a similar silhouette score (0.03) as Agglomerative clustering, indicating a comparable degree of overlap between clusters. However, Birch achieves better ARI, homogeneity, accuracy, and purity scores than Agglomerative clustering, suggesting that it is more effective in capturing the underlying structure of the MNIST dataset.

The improved performance of Birch compared to K-Means and Agglomerative clustering can be attributed to its unique approach to handling large datasets, which involves building a tree structure that approximates the data and then clustering the tree nodes. This hierarchical method allows Birch to capture more intricate patterns in the data, resulting in better-defined clusters.

Regarding the n_clusters parameter, we chose to set it to 10 for K-Means, Agglomerative clustering, and Birch, where that option was available, since the MNIST dataset contains 10 distinct digit classes. By matching the number of clusters to the number of classes, we provide a fair comparison among these clustering algorithms and facilitate the construction of classifiers based on those clustering results. However, it is important to note that clustering algorithms are inherently unsupervised, and setting the n_clusters parameter based on the number of classes effectively incorporated our prior knowledge about the data, which might not always be available or appropriate.

DBSCAN, on the other hand, does not have an n_clusters parameter and generated 27 clusters. The number of clusters was sensitive to the hyperparameters used (as well as to the underlying distribution of the data) since DBSCAN is a density-based algorithm, and it therefore likely detected smaller, denser subclusters within the digit classes, leading to a higher number of resulting clusters.

One additional note on the clustering algorithms is the comparison of the quantitative metrics and the visual inspection of the generated clusters, as in a few instances, the algorithm generated poor scores for measures like homogeneity and purity, but the visualized clusters appeared much more pure and homogeneous, and vice versa for others that had high scores for homogeneity and purity but had a mish-mosh of digits in the resulting clusters. This point warrants further investigation and inquiry, and we'd be thrilled to get some feedback about that!

Dimensionality Reduction: UMAP and Truncated SVD displayed the highest classification accuracies (93.1% and 91.4%, respectively) among the dimensionality reduction techniques, indicating an ability to preserve the intrinsic structure of the data more effectively. UMAP's performance can be attributed to its manifold learning approach, which seeks to preserve both local and global structures of the data and preserves the non-linear relationships between the features in the reduced space. Truncated SVD, on the other hand, works by truncating the singular value decomposition of the data matrix, retaining only the top k singular vectors, which are crucial for capturing the most significant patterns in the data.

PCA's performance (89.2%) likely benefited from its ability to capture the directions of maximum variance in the data, which often represent the most discriminative features. However, PCA's linear nature likely limited its capacity to capture non-linear relationships between features, explaining its lower accuracy compared to UMAP and Truncated SVD, as that limits the ability to differentiate between certain digit classes.

ICA and t-SNE, with accuracies of 82.2% and 85.3% respectively, underperformed compared to other dimensionality reduction techniques, because their primary goals do not necessarily align with the needs of the classification task at hand. ICA seeks to find statistically independent components in the data, which might not necessarily correspond to the most discriminative features for classifying the digit classes. t-SNE's primary focus on preserving local structure in the low-dimensional space may have compromised the characterization of the global structure, which would explain a lower classifier accuracy, as this loss of global structure can make it challenging to differentiate between certain digit classes that have similar local structures but differ in their global arrangement.

Stand-alone Classifiers: Random Forest's superior performance (97%) was likely attributed to its ability to model complex relationships between features by constructing an ensemble of decision trees. The method leverages the wisdom of the crowd, resulting in a more robust classifier and demonstrating its ability to model complex relationships between features. Logistic Regression, a linear model, performed reasonably well (93%) but was outperformed by the more flexible Random Forest classifier.

Conclusions: In conclusion, the disparity in performance among the various methods can be attributed to their unique characteristics and assumptions about the data. Birch, Agglomerative clustering, UMAP, and Truncated SVD performed well in their respective categories, and Random Forest emerged as the best stand-alone classifier for the MNIST digits. Further research could involve additional tuning of hyperparameters, exploring additional algorithms for the various tasks, and experimenting more with ensemble methods to improve the overall performance.