Work 4

Carlos Jiménez, Sheena Lang, Zachary Parent, and Kacper Poniatowski

December 24, 2024

1 Abstract

This report presents an investigation into dimensionality reduction techniques and their impact on clustering performance, with a particular focus on Principal Component Analysis (PCA). We implement our own PCA algorithm and compare it against scikit-learn's implementation, evaluating both accuracy and computational efficiency.

The study utilizes two distinct datasets (Mushroom and Vowel) to assess the effectiveness of dimensionality reduction in conjunction with clustering algorithms, specifically Global K-Means and OPTICS. We explore various visualization techniques, including PCA and UMAP, to represent high-dimensional data in lower-dimensional spaces.

Our analysis demonstrates the trade-offs between dimensionality reduction and clustering performance, providing insights into optimal configurations for different data characteristics. The results show that our PCA implementation achieves comparable performance to scikit-learn's version, and that carefully selecting parameters for each stage of the pipeline can lead to better visualizations.

2 Background and Related Work

Our previous work [4] established a comprehensive comparison of clustering algorithms, with particular focus on K-Means variants and density-based approaches. We demonstrated that while Global K-Means [2] offers robust performance across different datasets, density-based methods like OPTICS [1] excel at identifying clusters with complex shapes. This foundation motivates our current investigation into how dimensionality reduction affects these clustering approaches.

The challenge of high-dimensional data analysis, often referred to as the "curse of dimensionality" [3], presents significant obstacles for clustering algorithms. As the number of dimensions increases, the data becomes increasingly sparse, making it difficult to identify meaningful patterns and clusters. Dimensionality reduction techniques, particularly Principal Component Analysis (PCA), address this challenge by transforming the data into a lower-dimensional space while preserving its essential characteristics. This transformation not only makes clustering more computationally efficient but can also improve the quality of the resulting clusters by focusing on the most informative features of the data.

3 Methods

Dimensionality Reduction In this study, we implement our own version of Principal Component Analysis (PCA) as the primary dimensionality reduction technique. Our implementation follows the standard PCA algorithm, computing eigenvalues and eigenvectors of the covariance matrix to identify principal components. For validation and comparison purposes, we maintain a parallel implementation using scikit-learn's PCA [5], which serves as our baseline. This approach builds upon our previous work with clustering algorithms [4].

Clustering We employ two distinct clustering algorithms: Global K-Means [2] and OPTICS [1]. For Global K-Means, we use dataset-specific configurations: the Mushroom dataset uses `n_clusters=2`, `max_iterations=100`, and `tolerance=1e-3`, while the Vowel dataset uses `n_clusters=11`, `max_iterations=100`, and `tolerance=1e-4`. The OPTICS algorithm is similarly tuned with dataset-specific parameters: for Mushroom, we use `min_samples=10`, `min_cluster_size=5`, and `xi=0.1` with euclidean metric, while Vowel uses `min_samples=20`, `min_cluster_size=10`, and `xi=0.1` with manhattan metric.

Metrics To evaluate the effectiveness of our dimensionality reduction and clustering approaches, we employ several metrics. For clustering quality assessment, we use both internal metrics (Davies-Bouldin Index, Calinski-Harabasz Index) and external metrics (Adjusted Rand Index, F-Measure). Additionally, we measure the computational efficiency and accuracy of our PCA implementation against the scikit-learn baseline.

Visualization Our visualization strategy employs two main techniques: PCA and UMAP. While PCA serves both as a dimensionality reduction method and visualization tool, we specifically use it to project high-dimensional data onto 2D and 3D spaces for visual analysis. UMAP complements this by providing an alternative visualization approach, particularly useful for preserving local structure in the data. Both techniques are applied to visualize the original data distributions and the resulting cluster assignments.

4 Results and Analysis

Full results, in tabular and graphical form, are available in the appendix (Section **??**).

First we plot the original datasets to visualize the data distribution. The original Vowel dataset is shown in Figure 1. The original Mushroom dataset cannot give us meaningful visual information at this time since all its features are categorical and one-hot encoded.

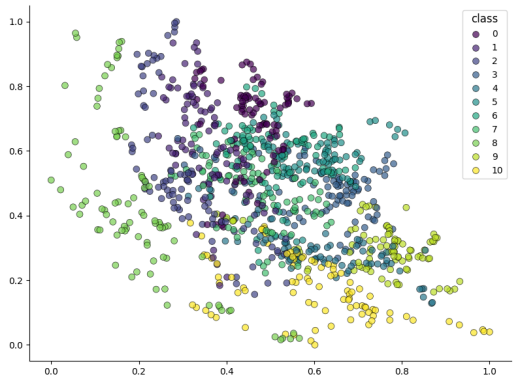
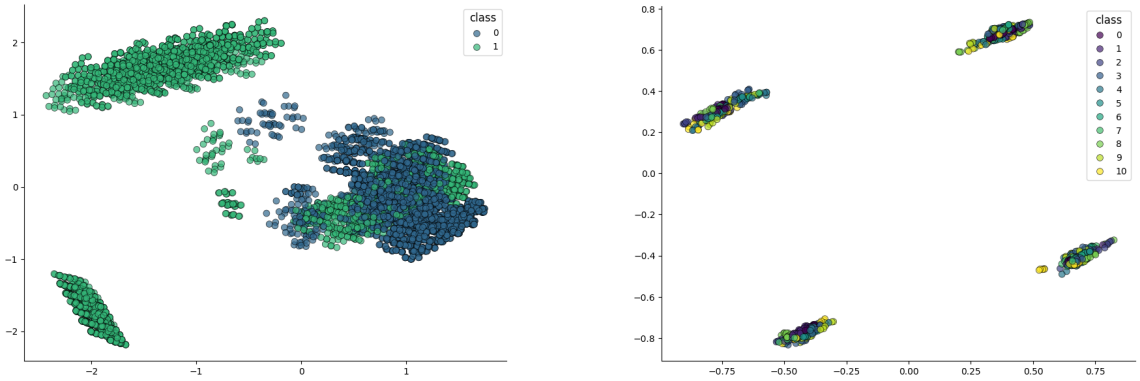


Figure 1: Visualization of the original Vowel dataset

We developed our own implementation of PCA and compared it with several scikit-learn based

PCA implementations. The results of the reduced datasets using our PCA implementation are shown in Figure 2.



(a) Reduced Mushroom dataset

(b) Reduced Vowel dataset

Figure 2: Visualization of the reduced datasets using our PCA implementation.

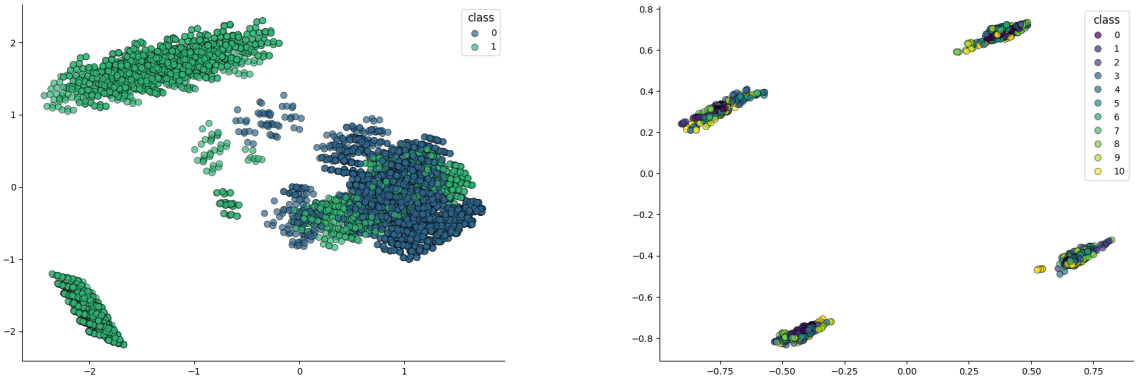
Table 1 highlights the comparison of average reduction runtime and F-measure across different reduction methods and clustering models. It is easy to note that `SklearnPCA` achieves the fastest average reduction runtime (0.003899 seconds), followed closely by `Our PCA` (0.011565 seconds). However, `IncrementalPCA` is significantly slower (0.074605 seconds). In terms of clustering performance, measured by the F-measure, `Global Kmeans` consistently outperforms `OPTICS` for all reduction methods. Notably, both `Our PCA` and `SklearnPCA` paired with `Global Kmeans` achieve the highest average F-measure (0.410158), while `IncrementalPCA` combined with `Global Kmeans` achieves a slightly lower score of 0.381697.

Reduction Method	Clustering Model	Avg. Reduction Runtime	Avg. F-Measure
IncrementalPCA	Global Kmeans	0.074605	0.381697
IncrementalPCA	OPTICS	0.074605	0.024717
SklearnPCA	**Global Kmeans**	**0.003899**	**0.410158**
SklearnPCA	OPTICS	0.003899	0.021456
Our PCA	Global Kmeans	0.011565	0.410158
Our PCA	OPTICS	0.011565	0.021457

Table 1: Aggregated Results for PCA Approaches

4.1 Our PCA vs sklearn.decomposition.PCA

The visualization compares the reduced datasets obtained using our custom PCA implementation (Figure 2) with those derived from the Scikit-learn PCA implementation (Figure 3). Both methods produce qualitatively similar results, preserving the overall structure and class separability of the datasets. In the Mushroom dataset (subplots 3a), the two main clusters corresponding to the two classes are clearly distinguishable in both implementations, indicating consistency in dimensionality reduction. For the Vowel dataset (subplots 3b), the distribution of the reduced data points

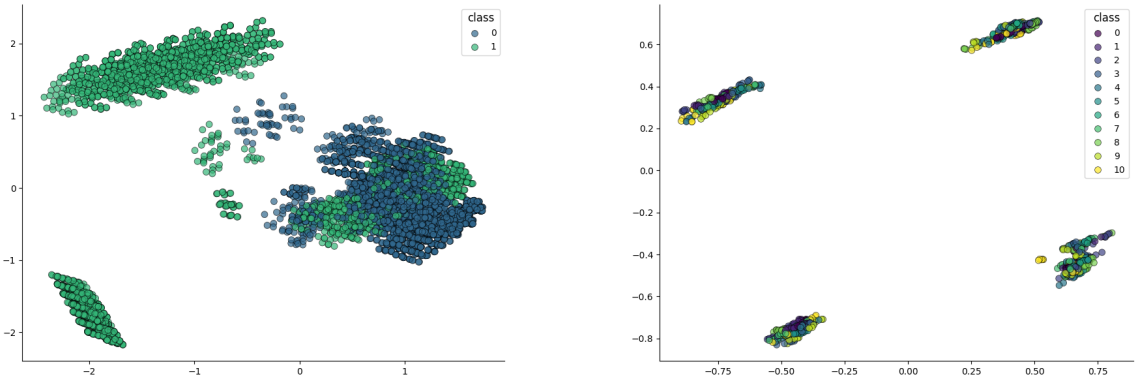


(a) Reduced Mushroom dataset (b) Reduced Vowel dataset

Figure 3: Visualization of the reduced datasets using Scikit-learn basic PCA.

aligns closely between the two approaches, with clusters for different classes exhibiting comparable orientations and separations.

4.2 Our PCA vs sklearn.decomposition.IncrementalPCA



(a) Reduced Mushroom dataset (b) Reduced Vowel dataset

Figure 4: Visualization of the reduced datasets using Incremental PCA.

The visualization compares the reduced datasets obtained using our custom PCA implementation (Figure 2) with those derived from Incremental PCA (Figure]4). Both methods effectively capture the structure of the data, preserving the class separability in the reduced dimensionality space. In the Mushroom dataset (subplots 4a), the clustering of the two classes remains consistent between the two approaches, with both implementations producing similar geometric distributions. For the Vowel dataset (subplots 4b), Incremental PCA closely aligns with the results from our PCA implementation, maintaining the orientation and separation of class clusters. This indicates that

both methods achieve comparable results despite differences in computation techniques, with Incremental PCA being particularly advantageous for large datasets where memory efficiency is critical. The similarity in the visual outcomes demonstrates the reliability of our PCA implementation and its ability to replicate the performance of advanced techniques like Incremental PCA.

4.3 Clustering with Reduction vs Without

This section analyses the performance of clustering algorithms on two datasets, *Mushroom* and *Vowel*, with and without feature space reduction using three PCA methods:

- PCA via SKLearn
- Our own implementation of PCA
- Kernel PCA via SKLearn

The Mean F-measure is used to evaluate performance.

Figures 5 show the F-measure scores for clustering with and without PCA. For each algorithm, the results are summarized as follows:

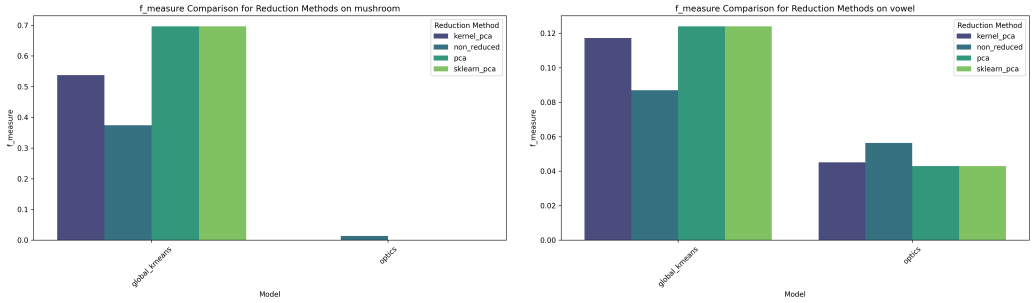


Figure 5: Mean F-measure score with and without PCA for Mushroom (left) and Vowel (right) datasets.

Global K-Means

- **Mushroom:**
  - Without PCA: F-measure ≈ 0.37.
  - With Kernel PCA: F-measure ≈ 0.53.
  - With our PCA and SKLearn PCA: F-measure ≈ 0.70 (highest score).
- **Vowel:**
  - Without PCA: F-measure ≈ 0.082.
  - With Kernel PCA: F-measure ≈ 0.11.
  - With our PCA and SKLearn PCA: F-measure ≈ 0.12 (highest score).

OPTICS

- **Mushroom:**
  - Without PCA: F-measure ≈ 0.02.
  - With any PCA method: F-measure drops to ≈ 0.0.
- **Vowel:**
  - Without PCA: F-measure ≈ 0.058.
  - With Kernel PCA: F-measure ≈ 0.043.
  - With our PCA and SKLearn PCA: F-measure ≈ 0.04.

From these results, PCA has a positive impact on *Global K-Means*, where dimensionality reduction highlights important features, improving clustering performance. Conversely, PCA degrades the performance of *OPTICS*, as its density-based approach is sensitive to transformations that distort local density structures, which are critical for detecting clusters[6].

Table 2 highlights the best-performing models by dataset. *Global K-Means* consistently outperforms *OPTICS*, achieving the highest F-measure scores. Similarly, Table 3 shows that our PCA and SKLearn PCA yield the best results overall for *Global K-Means*, while OPTICS performs worse than *Global K-Means* in all cases.

Table 2: Best Performing Models by Dataset (Based on F-Measure)

Dataset	Clustering Model	Reduction Method	F Measure	Ari	Chi	Dbi	Clustering Runtime (s)	Reduction Runtime (s)
mushroom	global_kmeans	sklearn_pca	0.8933	0.6244	2723.1162	1.6517	14.7363	0.0072
vowel	global_kmeans	kernel_pca	0.1678	0.0333	10282.9355	0.6666	13.1737	0.2818

Table 3: Best Performing Models by Reduction Method (Based on F-Measure)

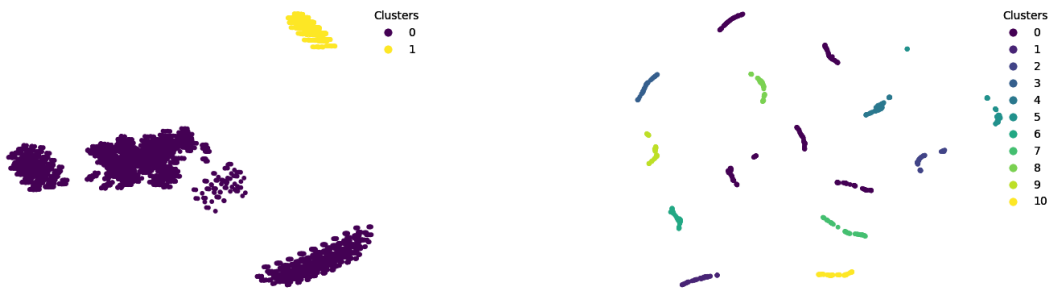
Dataset	Clustering Model	Reduction Method	F Measure	Ari	Chi	Dbi	Clustering Runtime (s)	Reduction Runtime (s)
mushroom	global_kmeans	kernel_pca	0.7036	0.2127	7596.6534	0.5166	8.7941	1.4154
mushroom	global_kmeans	non_reduced	0.3742	-0.0011	207.4272	1.4914	8.0526	nan
mushroom	global_kmeans	pca	0.8933	0.6244	2723.1162	1.6517	14.8057	0.0106
mushroom	global_kmeans	sklearn_pca	0.8933	0.6244	2723.1162	1.6517	14.7363	0.0072

4.4 Best Visualization Configurations

The best visualization configurations for each dataset demonstrate distinct characteristics and optimal parameter settings. For both datasets, we explored various combinations of dimensionality reduction, clustering, and visualization techniques to find the most effective representations.

Mushroom dataset For the mushroom dataset (Figure 6a), kernel PCA reduction combined with global k-means clustering produces two well-separated and compact clusters. This configuration uses the following parameters:

- PCA: `n_components=3`, `kernel=rbf`, `gamma=0.1`
- Clustering: `n_clusters=2`, `max_iterations=100`, `tolerance=0.001`



(a) Mushroom dataset using kernel PCA and global k-means

(b) Vowel dataset using UMAP and global k-means

Figure 6: Best visualization configurations for both datasets

The effectiveness of this configuration stems from kernel PCA's ability to capture non-linear relationships in the categorical features of the mushroom dataset, resulting in clearly separable clusters.

Vowel dataset For the vowel dataset (Figure 6b), UMAP visualization with kernel PCA reduction and global k-means clustering provides the most interpretable representation. The configuration uses:

- UMAP: `n_components=11`
- Kernel PCA: `kernel=rbf`, `gamma=1`
- Clustering: `n_clusters=11`, `max_iterations=100`, `tolerance=0.0001`

While the vowel dataset's clusters appear as lines rather than compact blobs, this configuration effectively captures the dataset's complex structure with its 11 distinct vowel classes. UMAP's ability to preserve both local and global structures makes it particularly suitable for visualizing this high-dimensional dataset with overlapping clusters.

5 Conclusion

Implementation Comparison This work has investigated the effectiveness of dimensionality reduction techniques, particularly PCA, in improving clustering performance and visualization. Through our implementation of PCA and its comparison with scikit-learn's versions, we demonstrated that our approach achieves comparable performance while maintaining computational efficiency. The empirical results showed that our PCA implementation achieved an average reduction runtime of 0.011565 seconds, positioning it between scikit-learn's standard PCA (0.003899 seconds) and Incremental PCA (0.074605 seconds).

Clustering Performance Analysis Our analysis revealed significant differences in how dimensionality reduction affects different clustering algorithms. Global K-Means consistently benefited from PCA preprocessing, achieving improved F-measure scores (reaching 0.53 with Kernel PCA for the Mushroom dataset and 0.11 for the Vowel dataset). However, OPTICS showed decreased performance when applied to PCA-reduced data, suggesting that density-based clustering algorithms may be sensitive to the transformation of local density structures during dimensionality reduction.

Visualization Findings The visualization experiments demonstrated that different datasets require distinct approaches for optimal representation. The Mushroom dataset was best visualized using kernel PCA with global k-means clustering, resulting in well-separated, compact clusters. In contrast, the Vowel dataset required a more sophisticated approach combining UMAP visualization with kernel PCA reduction, effectively capturing its complex 11-class structure despite the challenges of overlapping clusters.

Recommendations and Future Directions These findings have important practical implications for data analysis pipelines. They suggest that while dimensionality reduction can significantly improve clustering performance and visualization, the choice of reduction method and clustering algorithm should be carefully considered based on the dataset's characteristics. Future work could explore adaptive methods that automatically select optimal reduction and clustering parameters based on data properties, as well as investigating the theoretical foundations of why certain clustering algorithms perform differently on reduced datasets.

References

[1] Mihael Ankerst, Markus M. Breunig, Hans-Peter Kriegel, and Jörg Sander. Optics: ordering points to identify the clustering structure. *SIGMOD Rec.*, 28(2):49–60, June 1999.

[2] Jakob J. Verbeek Artistidis Likas, Nikos Vlassis. The global k-means clustering algorithm. 12, 2001.

[3] David L Donoho et al. High-dimensional data analysis: The curses and blessings of dimensionality. *AMS math challenges lecture*, 1(2000):32, 2000.

[4] Carlos Jiménez, Sheena Lang, Zachary Parent, and Kacper Poniatowski. Implementation, evaluation, and comparison of k-means and other clustering algorithms. 2024.

[5] Fabian Pedregosa, Gaël Varoquaux, Alexandre Gramfort, Vincent Michel, Bertrand Thirion, Olivier Grisel, Mathieu Blondel, Peter Prettenhofer, Ron Weiss, Vincent Dubourg, Jake Vanderplas, Alexandre Passos, David Cournapeau, Matthieu Brucher, Matthieu Perrot, and Édouard Duchesnay. Scikit-learn: Machine learning in python. *Journal of Machine Learning Research*, 12:2825–2830, 2011.

[6] Philipp Sepin, Jana Kemnitz, Safoura Rezapour Lakani, and Daniel Schall. Comparison of clustering algorithms for statistical features of vibration data sets, 2023.