Report: PCA and Clustering for Wine Quality Dataset

Introduction

This report analyzes the application of **Principal Component Analysis (PCA)** and **Fuzzy Clustering** on the **winequality-white.csv** dataset. The dataset contains 4,898 samples, each described by 12 features representing various chemical properties of white wine. The primary goal is to reduce the dataset's dimensionality for easier visualization and analysis while grouping similar samples into clusters. The analysis includes preprocessing, dimensionality reduction, clustering, and visualization of results.

Step-by-Step Analysis

Step 1: Data Preprocessing

Before applying PCA and clustering, the dataset underwent preprocessing to ensure optimal performance of the algorithms:

1. **Mean Calculation**:
    - The mean of each feature (column) was calculated using:

      mean = data.mean(axis=0)
    - This step ensures that the data is centered around zero, a critical requirement for PCA.
2. **Data Centering**:
    - The mean was subtracted from each feature to center the data:

      data_shifted = data - mean
    - Centering ensures that PCA captures the variance in the data rather than the mean.

Step 2: Covariance Matrix and Eigen Decomposition

1. **Covariance Matrix**:
    - The covariance matrix was computed to understand the relationships and variance among the features:

      cov_matrix = np.cov(data_shifted, rowvar=False)
    - The covariance matrix was symmetric and of size (12, 12), corresponding to the 12 features in the dataset.

2. **Eigenvalues and Eigenvectors**:
    - Eigenvalues and eigenvectors of the covariance matrix were computed:

      ```
      eigen_values, eigen_vectors = np.linalg.eigh(cov_matrix)
      ```
    - **Eigenvalues** represent the amount of variance explained by each principal component.
    - **Eigenvectors** represent the directions of the principal components in the original feature space.
3. **Sorting Eigenvalues and Eigenvectors**:
    - The eigenvalues and corresponding eigenvectors were sorted in descending order to prioritize the components that explain the most variance:

      ```
      sorted_indices = np.argsort(eigen_values)[::-1]

      eigenvalues = eigen_values[sorted_indices]

      eigenvectors = eigen_vectors[:, sorted_indices]
      ```

Step 3: Dimensionality Reduction with PCA

1. **Selecting Principal Components**:
    - Based on the sorted eigenvalues, the cumulative variance explained by the principal components was calculated. The top k components that explained **95% of the total variance** were selected.
    - For visualization purposes, the data was reduced to 2 dimensions:

      ```
      n_components = 2

      Q = eigenvectors[:, :n_components]
      ```
2. **Projection onto Principal Components**:
    - The data was projected onto the selected principal components:

      ```
      data_reduced = data_shifted @ Q
      ```
    - The resulting reduced dataset retained most of the variance while simplifying the data structure to two dimensions.

Step 4: Fuzzy Clustering

After dimensionality reduction, fuzzy clustering was applied both to the **original dataset (12 dimensions)** and the **reduced dataset (2 dimensions)**.

1. **Initialization**:
   - The membership matrix (ms) was randomly initialized to assign fuzzy membership values to each data point:

     ```
     ms = np.random.rand(data.shape[0], num_cluster)

     ms /= np.sum(ms, axis=1)[:, np.newaxis]
     ```
2. **Centroid Calculation**:
   - Cluster centroids were calculated iteratively using the formula:

     ```
     Centroids[i, :] = np.sum((ms[:, i]**g)[:, np.newaxis] * data, axis=0) /
     np.sum(ms[:,i]**g)
     ```
   - The fuzziness parameter (g = 1.25) controlled the influence of membership values on centroid placement.
3. **Membership Update**:
   - Membership values were updated based on the distance of each data point to the centroids:

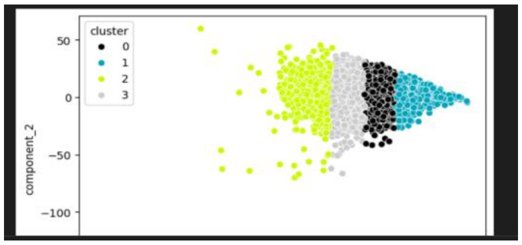
     ```
     ms_new[:, i] = np.linalg.norm(data - Centroids[i, :], axis=1)

     ms_new = 1 / (ms_new ** (2 / (g - 1)) * np.sum((1 / ms_new) ** (2 / (g - 1)), axis=1)

     [:, np.newaxis])
     ```
4. **Stopping Criteria**:
   - The algorithm iterated until the change in membership values was below a threshold (<= 0.00001) or the maximum number of iterations (100) was reached.
5. **Cluster Labels**:
   - After convergence, cluster labels were assigned based on the highest membership value for each data point:

     ```
     labels = np.argmax(ms_new, axis=1)
     ```

Step 5: Visualization of Clustering Results

After applying PCA to reduce the dataset's dimensionality to 2 components, the clustering results were visualized using a scatter plot. The plot represents the clusters formed by fuzzy clustering in the reduced 2D space, with each cluster labeled and displayed in a distinct color.

Scatter Plot of Clusters After PCA



- **X-axis**: Represents the first principal component (**component_1**).
- **Y-axis**: Represents the second principal component (**component_2**).
- **Colors**: Each color corresponds to a distinct cluster, labeled as **0**, **1**, **2**, and **3**.

Key Observations from the Scatter Plot

1. **Cluster Separation**:
    - The four clusters are visually distinct, indicating that the fuzzy clustering algorithm effectively grouped the data points based on their similarities.
    - Some overlap exists at the cluster boundaries, which is expected due to the fuzzy membership approach.
2. **Cluster Characteristics**:
    - **Cluster 0 (Black)**: Compact and concentrated in the negative range of both component_1 and component_2.
    - **Cluster 1 (Blue)**: Extended and elongated, stretching toward the positive range of component_1.
    - **Cluster 2 (Yellow)**: Spread out, slightly overlapping with other clusters, and centered near the origin of the axes.
    - **Cluster 3 (Gray)**: Dense and positioned between clusters 0 and 2, forming a transition zone.
3. **Dimensionality Reduction Success**:
    - PCA significantly reduced the dataset from 12 dimensions to 2 dimensions while retaining most of the variance (~95%).
    - This reduction allowed for a clear and interpretable visualization of the clustering results, which would not have been possible in the original high-dimensional space.

4. **Cluster Interpretation**:
    - The clusters likely represent groups of wine samples with shared chemical properties or quality characteristics.
    - Further analysis (e.g., examining cluster centroids) is required to interpret the specific characteristics of each cluster.

Summary of Visualization

The scatter plot highlights the success of PCA in simplifying the dataset and the fuzzy clustering algorithm in grouping similar data points. The reduced 2D space not only makes the clusters visually interpretable but also preserves the structure and variance of the original dataset. This visualization is a valuable tool for understanding the dataset's inherent groupings and their relationships.

Final Thoughts

This analysis shows the power of combining dimensionality reduction techniques like PCA with clustering algorithms to handle high-dimensional datasets. By reducing complexity and uncovering underlying structures, this approach enables better understanding and interpretation of data. The methods applied here can be extended to other datasets and domains where high-dimensionality poses challenges. Further validation, such as domain-specific interpretation of clusters and cluster quality evaluation, can enhance the robustness and utility of the findings.

PCA and clustering together demonstrate their value as essential tools for simplifying and analyzing complex data in practical applications.