University of Cape Town

Multivariate Statistics

Comparing Multivariate Embedding Methods for Plant Species Identification

Author:
Andomei Smit

Student Number:
SMTAND051

April 3, 2025

Contents

1 Introduction

Automatic plant identification from images has greatly benefited many industries including Forestry and Environmental Monitoring [1], Medicinal Plant Research [3] and Agriculture and Crop Management [5]. The sheer number of plant species (estimated at around 374 000 in 2016 [2]) and the complexity of accurate identification blend to create the perfect problem for advanced computer vision, machine learning and classification applications. Significant progress has already been made in this field.

Mallah et.al [4] attempted to find a classification method that could cope well with a small training set, a comparatively large number of species and the possibility of incomplete feature extraction. Using a sample of 1600 images of plant leaves corresponding to 100 unique species each represented by 16 different images, they extracted shape (representing the leaf contour), margin (representing the leaf edge) and texture (representing the internal patterns of the leaf) features. The images were taken of leaves of plants from the Royal Botanic Gardens, Kew, in the United Kingdom. A black-and-white version of the original images is shown in Figure 1.

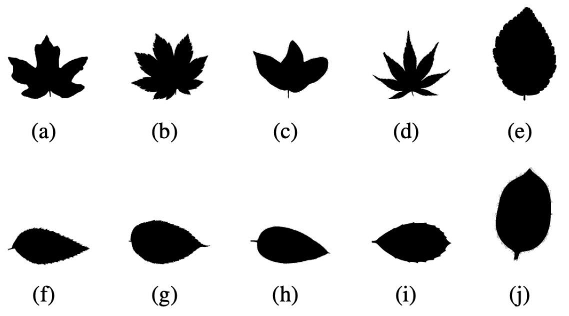

Figure 1: An example of 10 leaf images included in the data after being converted into binary (black-and-white) images. The binary version of the images were used to extract the shape and margin features, whereas a greyscale version was used to extract the texture feature.

This project will investigate whether the results of Mallah et al. could be improved upon by first using three different dimension reduction techniques to extract lower dimensional embeddings of the feature vectors. The techniques will be Principal Component Analysis (PCA), Isometric Mapping (Isomap) and Locally Linear Embedding (LLE). It will compare the performance of the K-Nearest Neighbors (K-NN) algorithm using Euclidean distance metrics when applied to the various lower dimensional representations of the features to the results from Mallah et al.

The data will be the extracted features from the leaf images that will serve as the independent variables with the species label (a unique label for each species) as the dependent categorical variable. Each feature is made up of 64 $1 \times n$ vectors, where n is the number of images. In total there are 192 feature vectors (64×3) across the three features.

2 Dimension Reduction

In order to reduce the dimensions of the original data, three dimension reduction techniques was be applied to the data, namely PCA, Isomap and LLE. This section gives a brief overview of each technique, explains the parameter tuning approach taken as well as the resulting tunings that were kept for further testing during the clustering stage.

Each of these techniques was applied to the 'Analysis Ready Dataset', which is the resulting data after the data cleaning, described in Section 4.1.

2.1 Principal Component Analysis

2.1.1 Overview of PCA

Principal Component Analysis (PCA) can be used as a linear dimension reduction technique to project high dimensional data into a lower dimensional space, while preserving as much variance as possible.

PCA operates by computing the eigenvectors and eigenvalues of the correlation matrix (used here since the data for this project is scaled and mean-centered). The eigenvectors are used to calculate Principal Components by projecting the original feature vectors onto the first k principal components. k can take on any value from 1 to 192, where 192 represents the total dimensions of the original data. This parameter requires tuning to achieve an appropriate balance between dimensionality and information retention.

Due to the high dimensionality of the data in this project (with 192 feature vectors), it is unlikely that the data is linear. However, PCA will be used as a baseline to compare the other two nonlinear methods to.

2.1.2 Parameter tuning

To determine a suitable number of Principal Components, k, the percentage of the total variance explained by the k^{th} eigenvector is plotted in Figure 2 in order to find an "elbow" point beyond which additional components contribute marginally to explaining the variance.

There seems to be a slight elbow at $k = 4$ and a more subtle bend at $k = 10$. However, the cumulative variance explained by the first 4 and 10 Principal Components is only 48% and 65%, respectively. In order to balance the elbow heuristic with the objective of retaining more of the total variance explained, a third value of k will be considered, namely $k = 30$, which accounts for 84% of the total variation in the data. This number of principal components corresponds to the Kaiser Criterion that selects all principal components with eigenvalues greater than 1. This criterion is applicable here since the PCA is applied to correlation matrix of standardized data.

Thus, the first 4, 10 and 30 principal components will be used to cluster the leaf species in the next section.

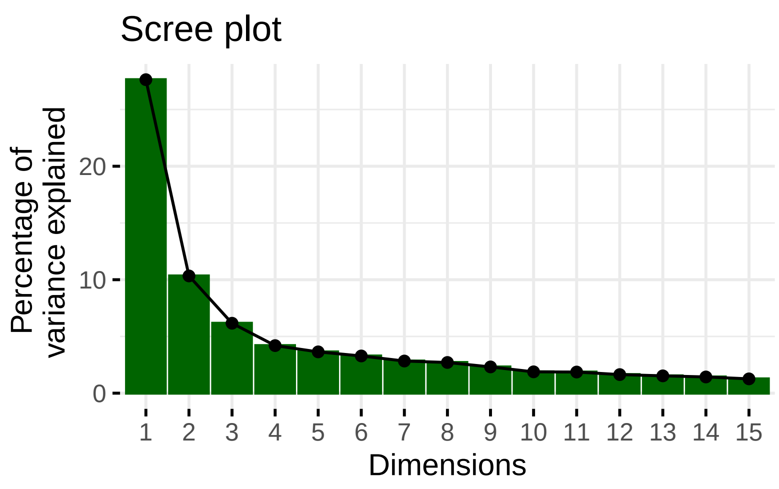

Figure 2: A scree plot of the percentage of variance explained by the k^{th} eigenvector. There seems to be a slight elbow at $k = 4$ and another at $k = 10$.

2.2 Isometric Mapping

2.2.1 Overview of Isomap

Isometric Mapping (Isomap) is a nonlinear dimension reduction technique that can be broken into three simple steps:

1. Construct a neighborhood graph by connecting the k nearest neighbor points
2. Calculate the geodesic distance between all sets of points as the distance of the shortest path between the points along the neighborhood graph
3. Apply classical Multidimensional Scaling (MDS)

2.2.2 Parameter tuning

For Isomap, the number of neighbors to use to construct the neighborhood graph, k, and the dimensions that the lower dimensional embedding should take on after applying MDS, d, needs careful tuning. To achieve an optimal combination of k and d, the residual variance for each combination should be minimized, where the residual variance is defined as

$$\text{Residual Variance} = 1 - R^2 = 1 - \text{cor}^2(D_G, D_Y) \tag{1}$$

where D_G is the matrix of the geodesic distances between all pairs of observations in the original higher dimensional space, D_Y is the matrix of Euclidean distances between all pairs of observations in the lower dimensional space.

The correlation between these two distance matrices represents how closely related the two distance matrices are and taken one minus this correlation coefficient thus represents the amount of the geodesic distance structure not captured by the lower dimensional embedding. Minimizing the residual variance is thus equivalent to finding the lower dimensional embedding that best preserves the original geodesic structure.

This is of course not the end goal for the embedding since the main aim is to find the embedding that eventually provides the best clustering accuracy. Thus, a set of best candidate combinations of k and d will be presented to investigate the overall clustering accuracy.

The parameter tuning approach was broken into four steps, namely:

1. Find the smallest value of k, call this k_{min}, for which the manifold is fully connect
2. Set a range of values for k (from k_{min} to some upper bound) and d and apply the Isomap algorithm to every combination of the parameters
3. Calculate the residual variance for each iteration above
4. Determine some candidate combinations of k and d that can be used for Isomap implementations to be tested in the clustering section

Step 1 is crucial as a one of the requirements for Isomap is that the neighborhood graph is fully connected, i.e. that each point in the data is reachable by any other point via the neighborhood graph. If this condition is violated, it means that the algorithm cannot determine the shortest distance along the graph to all points and the geodesic distances become undefined.

After applying the tuning steps above, the number of nearest neighbors for which the graph is fully connected, k_{min}, was found to be 13. Thus, a range for k and d was defined as follows:

$$k = 16 + 5i, \quad i \in \{0, 1, \ldots, 7\} \tag{2}$$

$$d = 2 + 10j, \quad j \in \{0, 1, \ldots, 7\} \tag{3}$$

For each combination of k and d, Isomap was applied and the residual variance calculated. Figure 3 plots the residual variance against the number of dimension for each k (left) and against the number of neighbors for each d (right). Plotting both k and d in this way enables the interpretation of the effect of changes in d and k on the residual variance.

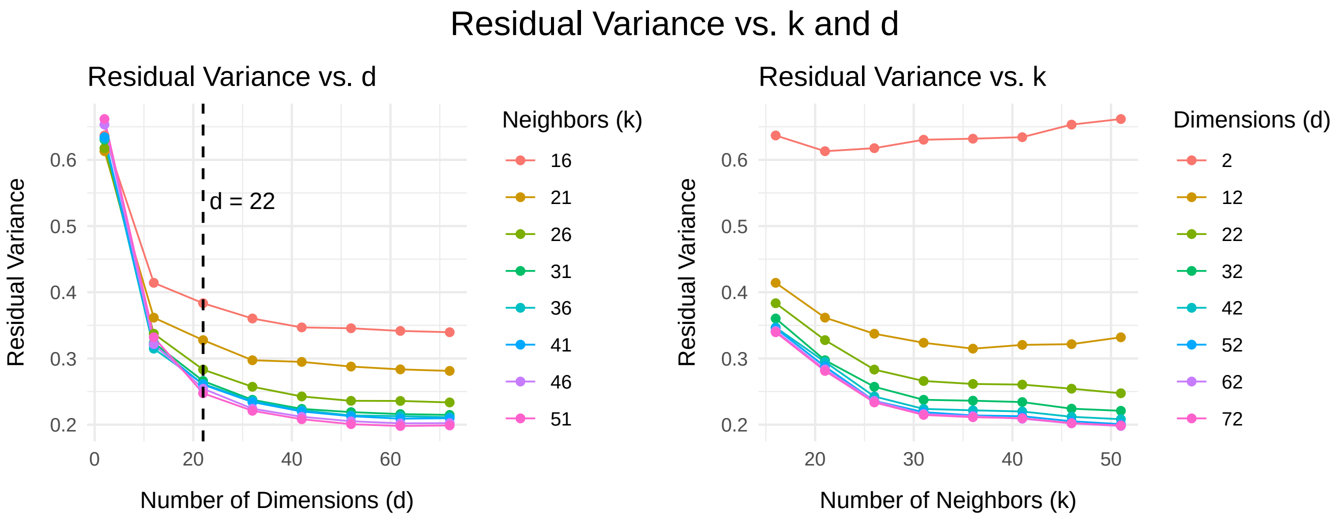

Figure 3: A plot of residual variance against the number of dimensions for the lower dimensional embedding, d (left), and the number of nearest neighbors, k (right) after applying the Isomap algorithm. The lines in the plot to the right represent different values of k and to the left different values of d.

As shown by the black dotted line on the plot of residual variance against d (left), past $d = 22$ dimensions used for MDS, the residual variance does not decrease substantially for any value of k. In other words, moving from $d = 12$ to $d = 22$ does show some improvement in the residual variance (i.e. a decrease), but beyond the elbow at $d = 22$, the line flattens out for all values of k. Thus, setting d to a value of 22 seems reasonable.

Next, the value for k can be investigated by looking at the plot of residual variance against k (right) and finding the green line corresponding to $d = 22$. This plot again confirms the observation first mentioned, i.e. that there is clearly a sharp decrease in residual variance from $d = 2,\ 12$ and 22 (although smaller when going from 12 to 22), but past this point the lines are almost equivalent at each value for k for higher values of d. It is also noted that there is only a slight reduction in variance for $k = 16,\ 21$ and 26. These are 0.38, 0.33 and 0.28, respectively. The residual variance still reduces steadily past these points for larger k, but if k is too large, it would poorly estimate the geodesic distances as points very far apart would be considered neighbors. This leads to the shortest path along the neighborhood graph being roughly equal to the Euclidean distance, which is exactly what the algorithm tries to get away from.

Thus, the combinations of d and k that will be used to test the clustering algorithm are $\{d = 22, k = 16\}$, $\{d = 22, k = 21\}$ and $\{d = 22, k = 26\}$.

2.3 Locally Linear Embedding

2.3.1 Overview of LLE

Locally Linear Embedding (LLE) is a nonlinear dimension reduction technique that, like Isomap, relies on the concept of local neighborhoods. It begins by identifying the k nearest neighbors for each observation in the high-dimensional space. Each data point is then reconstructed as a linear combination of its neighbors, using a set of weights that best preserve the local geometry. These weights are constrained such that they sum to one and are nonzero only for the selected neighbors.

Once the reconstruction weights are computed, LLE seeks a lower-dimensional embedding of the data in which each point can be similarly reconstructed from the same weights and its corresponding neighbors. The optimal embedding is found by minimizing the overall reconstruction error in the low-dimensional space, thereby ensuring that the local neighborhood structure of the original data is maintained. For a more in-depth mathematical representation, see Section 4.2.

2.3.2 Parameter tuning

Similar to Isomap, the parameters k (the number of nearest neighbors) and d (the dimensions of the lower dimensional embedding) need careful tuning. A further restriction is placed on k in that k must be greater than d to ensure that the weights, w_{ij} are stable and not underdetermined. If $k \leq d$, Equation 11 is under-constrained with d unknowns and only k equations.

For this reason, a range of values for k and d is again defined as

$$k = 5 + 5i, \quad i \in \{0, 1, \ldots, 9\} \tag{4}$$

$$d = 2 + 10j, \quad j \in \{0, 1, \ldots, 7\} \tag{5}$$

where LLE is applied for each combination of the parameter sets. However, if $k \leq d$, the combination is ignored and the algorithm will skip to the next combination. For each combination, the reconstruction error is calculated. This is defined as the difference between the high and low dimensional embedding, or

$$\mathcal{E}(\mathbf{W}) = \sum_{i=1}^{n} \left\| \mathbf{x}_i - \sum_{j \in \mathcal{N}(i)} w_{ij} \mathbf{x}_j \right\|^2 \tag{6}$$

The aim was to minimize this error, thus to preserve the local geometry in the high dimensional space. For each valid combination of k and d, i.e. where $k > d$, the reconstruction error is plotted in Figure 4. It is clear that the reconstruction error is relatively constant for a set d across different values of k. Since the ultimate goal is not just to minimize the reconstruction error, but to eventually achieve the highest clustering accuracy, higher values for k were chosen for $d = 2$, but for the other dimensions, the lowest value of k for each d was chosen to be used in the classification step. Although the reconstruction error is lowest at $d = 2$, higher dimensions will also be tested since the original data is image data, which lends itself to be better represented by higher dimensions.

Thus, the combinations of d and k that will be used to test the clustering algorithm are $\{d = 2, k = 15\}$, $\{d = 12, k = 15\}$, $\{d = 22, k = 25\}$, $\{d = 32, k = 35\}$ and $\{d = 42, k = 45\}$.

2.4 Summary of all dimension reduction techniques

In summary, the K-Means clustering algorithm will be applied to the each of the following configurations for each dimension reduction technique:

1. PCA: 4, 10 and 30 principal components (call these pca_4, pca_10 and pca_30, respectively).
2. Isomap: $\{d = 22, k = 16\}$, $\{d = 22, k = 21\}$ and $\{d = 22, k = 26\}$ (call these iso_k16_d22, iso_k21_d22 and iso_k26_d22, respectively).
3. LLE: $\{d = 2, k = 15\}$, $\{d = 12, k = 15\}$, $\{d = 22, k = 25\}$, $\{d = 32, k = 35\}$ and $\{d = 42, k = 45\}$ (call these lle_k15_d2, lle_k15_d12, lle_k25_d22, lle_k35_d32 and lle_k45_d42, respectively).

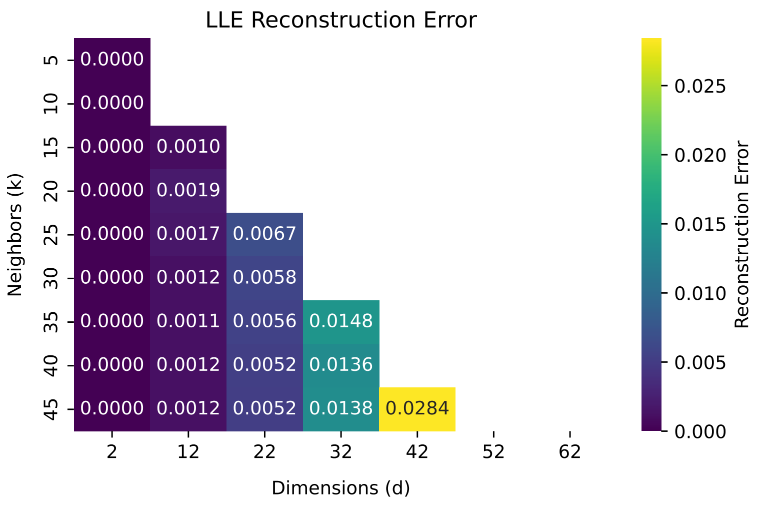

Figure 4: A plot of the reconstruction error after applying LLE for different values of k (y-axis) and d (x-axis). It is clear that along the columns, i.e. for set d, the reconstruction error does not vary much for different values of k.

Since this is not an unsupervised learning problem because the species classification is known, the only measure of clustering success that will be used is clustering accuracy. This will be compared to the results from Mallah et.al. [4] to see if dimension reduction could improve their clustering accuracy.

3 Clustering with K-Means

3.1 Implementation

In order to evaluate the ability of the various dimension reduction techniques to improve the clustering accuracy, K-Means with random initial cluster centroids was applied for 100 different random starting centroids. The accuracy at each centroid was evaluated as the number of species that were correctly classified. The overall accuracy was taken as the mean accuracy across the 100 iterations. The mean accuracy and various other performance metrics were calculated and can be seen in Table 1.

One of the problems with applying the K-means algorithm is that the cluster labels are assigned arbitrarily in each round, meaning that the cluster numbers themselves do not carry any meaning. In order to evaluate the clustering accuracy, the cluster labels needed to be mapped to the Species labels at each round.

To achieve this, the Hungarian algorithm (also known as the Kuhn-Munkres algorithm) was applied. The algorithm works by building a confusion matrix between the true Species labels and cluster labels and finding the best one-to-one mapping that maximizes the total correct assignments.

Table 1: Average performance metrics after applying the K-Means algorithm to each reduced-dimension dataset across 100 iterations. Labels refer to the dimension reduction technique and its parameters (see Section 2.4). SD indicates standard deviation.

Dataset	**Accuracy (%)**	**SD (%)**	**Silhouette**	**Silhouette SD**	**Time (s)**
Original (ard1)	78.77	2.35	0.22	0.00	225.0
PCA (4 PCs)	56.89	1.63	0.31	0.01	11.1
PCA (10 PCs)	76.79	1.84	0.33	0.01	18.1
PCA (30 PCs)	78.82	2.11	0.30	0.01	29.0
Isomap ($k = 16$, $d = 22$)	76.66	1.70	0.35	0.00	21.6
Isomap ($k = 21$, $d = 22$)	75.68	1.84	0.31	0.00	22.6
Isomap ($k = 26$, $d = 22$)	74.00	1.79	0.29	0.00	23.9
LLE ($k = 15$, $d = 2$)	30.01	0.63	0.55	0.01	5.6
LLE ($k = 15$, $d = 12$)	62.31	1.88	0.44	0.01	17.7
LLE ($k = 25$, $d = 22$)	66.04	2.13	0.38	0.01	32.3
LLE ($k = 35$, $d = 32$)	74.17	2.10	0.38	0.01	52.4
LLE ($k = 45$, $d = 42$)	77.95	1.85	0.38	0.01	67.8

3.2 Discussion

From Table 1 the highest mean accuracies from each algorithm was pca_30 for PCA, iso_k16_d22 for Isomap and lle_k45_d42 for LLE, with mean accuracies of 78.8%, 76.7% and 78.0%, respectively. However, when taking into account their standard deviations, these algorithms perform very similarly in terms of mean accuracy. It is further noted that the reduced dimension datasets for all three algorithms did not improve the overall accuracy from applying K-Means to the original data, ard1, that had a mean accuracy of 78.8% and standard deviation of 2.4%.

It is evident from the elapsed time that the runtime for both PCA and LLE increases substantially as the dimensionality increases. This supports the well-known limitation that K-Means does not scale well with high-dimensional data, as it requires computing a full distance matrix at each iteration—a process that is computationally intensive. This effect is most pronounced when applying K-Means directly to the original dataset, ard1, which results in the highest computational cost among all tested configurations.

The same conclusion cannot be drawn for Isomap, since all three configurations used

22 dimensions in the lower dimensional embeddings, but it is expected that the same phenomenon would be observed.

The silhouette scores give a combined insight into how well the images are clustered within a cluster and how distinct they are from images in other clusters. Here it can be seen that for PCA and Isomap, the silhouette scores are very similar for different configurations of the same algorithm (even taking into account their standard deviations which are very low across all the configurations).

This is true even for PCA where the clustering accuracy varies by more than 20% from pca_4 to pca_30. Thus, even when the clustering accuracy is high, the clusters are not very distinct or well clustered (as shown by the low silhouette scores).

This could be due to the original images being very similar (and thus not able to be clustered into distinct clusters, which could be said when looking at the silhouette score of ard1), the features extracted from the images not capturing the features (shape, margin and texture) well or even the dimension reduction techniques actually getting rid of differences between features that would have been key to distinguish between species.

3.3 Conclusion

Overall, none of the configurations of different dimension reduction algorithms produced a clustering outcome that could compete of that from Mallah et al. [4]. The dimension reduction algorithms were not even able to improve upon the result of using the full dataset to apply K-Means (other than reducing the computational time significantly).

There exists many more avenues that could be explored before coming to the conclusion that all of these dimension reduction techniques cannot improve the clustering algorithm. Below some of these avenues are briefly listed.

3.3.1 Alternative clustering algorithms

LLE and Isomap are non-linear dimension reduction techniques, meaning that apply K-Means (which is a linear dimension reduction technique) may not be the best choice of algorithm. Algorithms such as Kernel K-Means could be explored that allows for non-linear structures in the data.

3.3.2 Alternative approach to dimension reduction

In the approach of this paper all the features from shape, margin and texture were grouped together and the dimension reduction techniques applied to the full set.

Mallah et al. used a density based clustering method on each of these features individually before combining them into a final posterior probability to classify the clusters.

A similar approach could be taken here to reduce the dimensions of the three feature sets individually (i.e. for each set of 64 features), before applying a clustering algorithm to the dataset of the three combined reduced dimension sets from shape, margin and texture.

4 Appendix

4.1 Data Preparation

It was noted that of the 1600 images from 100 species, two of the species had incomplete feature data (i.e. were missing one of the shape, margin, or texture features for at least one of the samples of a species). These were originally left out by the authors to test the ability of their probability-based estimators when dealing with missing features.

However, since the images do not have a unique sample identifier across the extracted features, it was impossible to know which sample of that species was missing that feature vector. Thus, the entire species that is missing a feature for one of the samples needs to be removed. It was further assumed that the order of samples in the species was consistent across the three different feature sets (i.e., that the first sample for the species was the same first sample across all the features).

This project will thus only use the features extracted from the remaining 1568 images of 98 unique species. The final dimensions of the data are 1568 (images, or n) by 192 (features, or p). Each of the feature vectors was continuous vectors on a scale roughly between 0 and 0.2. The exact summary statistics for these features was omitted due to the limited insight they provide given the size, but Table 2 shows an example of the first two feature vectors for each feature for two samples of two different species. The details for this data preparation can be seen in the Initial Data Cleaning R Markdown file on GitHub.

Species	Margin Feature 1	Margin Feature 2	Shape Feature 1	Shape Feature 2	Texture Feature 1	Texture Feature 2
Acer Circinatum	0.00000	0.00000	0.00070	0.00072	0.02441	0.00098
Acer Rubrum	0.00195	0.00000	0.00000	0.00099	0.00101	0.00000

Table 2: The first two feature vectors for each Feature for two different Species. In total each Feature (Margin, Shape and Texture) had 64 features each (i.e., 64 columns), thus each sample has 192 feature measurements each.

It was further noticed that the scale of the feature vectors were not all consistent. This would cause an issue for any of the dimension reduction techniques in this project since they all rely (in some form or another) on various distance metrics and/or variance between observations to compute the lower dimensional embeddings. If observations are not on the same scale, some feature vectors could dominate the embeddings and skew the results in the lower dimensional space. In order to remediate this issue, the data were scaled to be of unit variance and mean-centered.

4.2 Mathematical configuration of LLE

Using more specific mathematical notation, the LLE algorithm can be summarized in three steps. These are:

1. Calculate the k nearest neighbors for all observations in the data
2. Calculate the set of weights, w_{ij} to best construct the original observations from their neighbors i.e. such that

$$\mathbf{x}_i \approx \sum_{j \in N(i)} w_{ij} \mathbf{x}_j \quad \forall\ i = 1, \ldots, N \tag{7}$$

   where $j \in N(i)$ represent the k observations in the neighborhood of observation i, subject to

$$\sum_j w_{ij} = 1 \tag{8}$$

$$w_{ij} = 0 \quad \forall\ j \notin N(i) \tag{9}$$

3. Find the low dimensional embedding $\mathbf{y}_i \in \mathbb{R}^d$ such that

$$\mathbf{y}_i \approx \sum_{j \in N(i)} w_{ij} \mathbf{y}_j \quad \forall\ i = 1, \ldots, N \tag{10}$$

   by minimizing the cost function

$$\Phi(\mathbf{Y}) = \sum_{i=1}^{N} \left\| \mathbf{y}_i - \sum_j w_{ij} \mathbf{y}_j \right\|^2 \tag{11}$$

   where

$$\mathbf{Y} \in \mathbb{R}^{n \times d} \tag{12}$$

   is the low dimensional embedded observations and w_{ij} are the set of weights from Step 2

4.3 Link to Github Repository

The full code and dataset used in this project are available on GitHub at: `https://github.com/Annie0619/multivariate_assignment`.

References

[1] M. A. Beck, C. Y. Liu, C. P. Bidinosti, C. J. Henry, C. M. Godee, and M. Ajmani. An embedded system for the automated generation of labeled plant images to enable machine learning applications in agriculture. *PLoS One*, 15(12):e0243923, 2020.

[2] Maarten J.M. Christenhusz and James W. Byng. The number of known plant species in the world and its annual increase. *Phytotaxa*, 261(3):201–217, 2016.

[3] Owais A. Malik, Nazrul Ismail, Burhan R. Hussein, and Umar Yahya. Automated real-time identification of medicinal plants species in natural environment using deep learning models—a case study from borneo region. *Plants*, 11(15):1952, 2022.

[4] Charles Mallah, James Cope, and James Orwell. Plant leaf classification using probabilistic integration of shape, texture and margin features. In *Proceedings of the IASTED International Conference on Signal Processing, Pattern Recognition and Applications (SPPRA)*, pages 105–110, 2013.

[5] Girma Tariku, Isabella Ghiglieno, Gianni Gilioli, Fulvio Gentilin, Stefano Armiraglio, and Ivan Serina. Automated identification and classification of plant species in heterogeneous plant areas using unmanned aerial vehicle-collected rgb images and transfer learning. *Drones*, 7(10):599, 2023.