Proceedings of the 2023 International Conference on Machine Learning and Automation
DOI: 10.54254/2755-2721/31/20230153

Evaluation of dimensionality reduction and unsupervised clustering methods in breast datasets

JunFan Liu[1,5,*], **Weide He**[2,6], **YuXin Wang**[3,7], **BoYi Zhang**[4,8]

[1] Kanazawa University, Kanazawa, Japan
[2] Nanjing University of Information Science and Technology, Nanjing, China
[3] Hangzhou Xuejun High School, Hangzhou, China
[4] Wuhan Britain-China School, Wuhan, China

[5] liujunfan@stu.kanazawa-u.ac.jp
[6] weldom888@hotmail.com
[7] 2969271700@qq.com
[8] belinda_zby@163.com
[*] corresponding author

Abstract. This paper delves into the issues related to handling high-dimensional data in massive datasets, such as computational challenges and uneven data distribution owing to diminished data point density. Various dimensionality reduction techniques such as Principal Component Analysis (PCA), Kernel Principal Component Analysis (KPCA), and Diffusion Maps are discussed and evaluated for their efficiency in extracting crucial data features. This aids in gaining a comprehensive understanding of the data. The study also examines unsupervised clustering methods like K-means, DBSCAN, and spectral clustering. By integrating these clustering methods with dimensionality reduction techniques, we aim to uncover potential synergies. The principles and methodology behind spectral clustering and unsupervised nonlinear diffusion learning are further dissected. Various datasets are employed to evaluate the efficiency of these techniques empirically. The final section of the paper comprises an evaluation of the clustering results and a discussion on potential avenues for future research.

Keywords: Dimensionality Reduction, Unsupervised Clustering, Machine Learning.

1. Introduction

The presence of high-dimensional data is a common phenomenon when dealing with large datasets. Such data can lead to computational challenges and uneven data distribution due to reduced data point density. In addition, acquiring labels for high-dimensional data points can be a costly affair without effective dimensionality reduction. Therefore, reducing dimensionality aids in better understanding and data exploration. Extracting useful information from data for analysis often requires a comprehensive understanding of the data. By aiding in key feature extraction, dimensionality reduction can enhance our data understanding and yield improved results.

Clustering methods, which divide a large set of unlabeled data into multiple categories based on inherent similarities, prove effective for reducing data dimensionality. In this report, we apply K-means, Mean-shift, and DBSCAN clustering algorithms.

DOI: 10.54254/2755-2721/31/20230153

Effective dimensionality reduction and clustering techniques are critical due to the complex and high-dimensional nature of data in various fields like bioinformatics, image processing, and natural language processing. Such techniques facilitate a better understanding and visualization of data structure and relationships and aid in predictions and discoveries. They are integral tools in data analysis and machine learning.

In this paper, we review several dimensionality reduction and clustering techniques, including Principal Component Analysis (PCA), Kernel Principal Component Analysis (kPCA), and Diffusion Maps, along with unsupervised clustering algorithms like K-means, Mean-shift, and DBSCAN. We also explore Spectral Clustering, a technique that combines clustering and dimension reduction concepts, bridging the gap between these two crucial fields.

We apply these techniques on the Breast Cancer dataset, commonly used in machine learning for binary classification tasks. The dataset, with its high dimensionality, is ideal for testing the techniques discussed in this study.

The report starts with a review of the research materials and methodologies, followed by an in-depth discussion of each dimensionality reduction method and their mathematical underpinnings. We then overview the clustering algorithms examined in this study. Subsequently, we detail the setup and results of our experiments and discuss the efficiency and performance of each method. The report concludes with a summary of our findings and suggestions for future research.

2. Methods and Materials

2.1. Data Dimensionality Reduction

Data dimensionality reduction is a vital technique in machine learning and data science for handling high-dimensional data. The goal of this process is to trim down the feature space by preserving the most valuable features and eliminating the less important ones. This approach can significantly speed up data processing. By reducing the number of data dimensions, we can optimize computational resources, filter out noise, and enhance the interpretability of extracted features.

2.1.1. Understanding Principal Component Analysis Principal Component Analysis (PCA) is a linear procedure that effectively reduces data dimensionality. It performs an orthogonal transformation, converting possibly correlated variable observations into a set of linearly uncorrelated variables called principal components. PCA is a key tool for analyzing multivariate statistical distributions with characteristic values. It can reveal the underlying structure of the data, aiding in a more insightful interpretation of data variables.

PCA is particularly insightful when dealing with a high number of samples. In PCA plots, samples with similar compositions tend to cluster together. PCA assesses the similarity of sample composition by examining the dispersion or aggregation patterns among the samples.

However, PCA has certain limitations. It requires a mean removal process before constructing the PCA covariance matrix, which can be problematic in specific fields where all signals should be non-negative. PCA also assumes the linearity of the data, so non-linear patterns can potentially skew the analysis. Moreover, the transformed variables might not be as interpretable as the original data since they are linear combinations of the initial variables. An improperly conducted PCA can also lead to significant information loss.

The PCA process starts with data standardization.

$$x = \frac{x - \text{mean}(x)}{\text{var}(x)} \tag{1}$$

Subsequently, the covariance matrix is calculated.

$$\Sigma = \frac{1}{n} \sum_{i=1}^{n} x^{(i)} x^{(i)^T} \tag{2}$$

DOI: 10.54254/2755-2721/31/20230153

Then, the covariance matrix undergoes an eigenvalue decomposition.

$$\Sigma = u\Lambda u^T \tag{3}$$

Once the eigenvalues and eigenvectors of the covariance matrix are obtained, the eigenvalues are sorted in descending order, the top k ones are selected, and the corresponding k eigenvectors are assembled into an eigenvector matrix. Finally, the data is mapped into a new space formed by these k eigenvectors.

2.1.2. Exploring Kernel Principal Component Analysis Kernel Principal Component Analysis (KPCA) extends PCA by employing a kernel method to handle non-linear separability in the dataset. In KPCA, the data is projected into a higher-dimensional space using a kernel function, making it possible to capture non-linear correlations between variables. KPCA essentially seeks a linear projection of the data in this higher-dimensional space that preserves the most variance.

KPCA is able to detect non-linear patterns in the data, which may remain hidden in the original feature space, providing a key advantage over traditional PCA. However, KPCA can be computationally demanding, especially for larger datasets, due to the calculation of the kernel matrix. Moreover, KPCA is sensitive to outliers, which can disrupt the computation of the kernel matrix and affect the extraction of principal components.

Kernel Functions KPCA uses various kernel functions, including:

- Gaussian kernel function:

$$G(x, y) = \exp\left(-\frac{|x - y|^2}{2s^2}\right) \tag{4}$$

These kernel functions give KPCA the ability to project the data into a higher-dimensional feature space where it can be projected onto the principal components and then linearly separated.

Gaussian Kernel Function The Gaussian kernel function, often referred to as a radial basis function (RBF) kernel, is a commonly used kernel function in KPCA. Given any two input vectors x_i and x_j, the Gaussian kernel function with parameter $sigma$ is defined as:

$$\gamma(x_i, x_j) = \exp\left(-\frac{|x_i - x_j|^2}{2s^2}\right) \tag{5}$$

2.1.3. Diffusion Mapping Diffusion mapping is a versatile technique for dimensionality reduction, particularly effective in addressing non-linear systems. It takes advantage of the data manifold's intrinsic geometric properties to form lower-dimensional representations [1]. The algorithm constructs embeddings in Euclidean space using the eigenvalues and eigenvectors of a diffusion operator applied to the data. The core idea is that the Euclidean distance within this embedded space reflects the "diffusion distance" in the original data points. In this section, we will elaborate on the essential mathematical concepts underpinning the diffusion mapping technique.

Understanding Euclidean Distance Assume that we have points in an n-dimensional Euclidean space, denoted by Cartesian coordinates $\alpha_1 \ldots \alpha_n$ and $\beta_1 \ldots \beta_n$. The distance between these points can be calculated as:

$$d(\alpha, \beta) = \sqrt{(\alpha_1 - \beta_1)^2 + (\alpha_2 - \beta_2)^2 + \cdots + (\alpha_n - \beta_n)^2} \tag{6}$$

The Euclidean distance can also be concisely depicted as the Euclidean norm of the vector difference between the two coordinates[2]:

$$d(\alpha, \beta) = |\alpha - \beta|. \tag{7}$$

DOI: 10.54254/2755-2721/31/20230153

Concept of Connectivity Connectivity denotes the transition probability from one data point, α, to another, β, in a single random walk step[3]:

$$\text{connectivity } (\alpha, \beta) = \pi(\alpha, \beta) \tag{8}$$

This probability can be defined relative to a kernel function, κ, termed the diffusion kernel:

$$\text{connectivity}(\alpha, \beta) \propto \kappa(\alpha, \beta) \tag{9}$$

The kernel function provides a local similarity measure, restricted to a specific neighborhood and fading beyond its limits. The matrix $\kappa \in \mathbb{R}^{N \times N}$ evaluates this similarity based on the Euclidean distance between data points using a Gaussian kernel:

$$\kappa_{\alpha,\beta} = \exp\left(-\frac{|\alpha - \beta|^2}{2\sigma^2}\right) \tag{10}$$

Here, σ determines the kernel's scope, with larger values leading to broader embeddings and smaller values highlighting local structures.

Interpreting Diffusion Distance The row-normalized diffusion matrix, Θ, has elements defined as:

$$\Theta_{ij} = \pi\left(\mathcal{X}_i, \mathcal{X}j\right) \tag{11}$$

Each matrix element, $\Theta i, j$, signifies the transition probability between data points i and j. The diffusion metric, derived from this matrix, quantifies the similarity between two points based on their connectivity[4]:

$$\mathcal{D}_t\left(\mathcal{X}i, \mathcal{X}j\right)^2 = \sum \upsilon \in \mathcal{X} \left|\pi_t\left(\mathcal{X}i, \upsilon\right) - \pi_t\left(\mathcal{X}j, \upsilon\right)\right|^2 = \sum \kappa \left|\Theta i\kappa^t - \Theta\kappa j^t\right|^2 \tag{12}$$

Constructing a Diffusion Map Diffusion mapping constructs a transformation between the data and diffusion spaces, effectively organizing the data in accordance with the diffusion metric. The map is formed as[1]:

$$\mathcal{Y}_i := \left[\begin{array}{c} \pi_t\left(\mathcal{X}_i, \mathcal{X}_1\right) \pi_t\left(\mathcal{X}_i, \mathcal{X}_2\right) \vdots \pi_t\left(\mathcal{X}_i, \mathcal{X}N\right) \end{array}\right] = \Theta i*^T \tag{13}$$

In this arrangement, the Euclidean distance between two mapped points, $\mathcal{Y}_i$ and $\mathcal{Y}_j$, resembles the diffusion distance between the original data points $\mathcal{X}_i$ and $\mathcal{X}_j$. The diffusion distances can be related to the eigenvalues and eigenvectors of Θ as:

$$\mathcal{Y}_i' = \left[\begin{array}{c} \omega_1^t\varphi_1(i)\omega_2^t\varphi_2(i) \vdots \omega_n^t\varphi_n(i) \end{array}\right] \tag{14}$$

The Euclidean distance between the transformed points $\mathcal{Y}_i'$ and $\mathcal{Y}_j'$ also mirrors the diffusion distance. Dimensionality reduction is carried out by retaining the m dimensions associated with the most significant eigenvectors, which provide the best approximation of the diffusion distance, $\mathcal{D}_t\left(\mathcal{X}_i, \mathcal{X}_j\right)$. Therefore, the diffusion map that optimally preserves the data's inherent geometry is $\mathcal{Y}_i'$.

2.2. Unsupervised Clustering

2.2.1. K-means The k-means algorithm aims to partition a cluster C and its respective centers z, in a manner that reduces the overall dissimilarity as much as possible [5]. The process involves four key steps:

Initially, center points must be established. This is done by randomly choosing k data points as the central points for each cluster. Here, k should fall between 0 and n (n symbolizing the total objects in the dataset). If k equals n, the setup becomes insignificant[6].

DOI: 10.54254/2755-2721/31/20230153

Subsequently, each point is allocated to the nearest cluster by calculating its Euclidean Distance as follows:

$$dist_e d(X,Y) = \sqrt{\sum_{i=1}^{m}(X_i - Y_i)^2} \tag{15}$$

Here, consider two samples each with m dimensions:

$$X = (x_1, x_2 ... x_m),\ Y = (y_1, y_2 ... y_m) \tag{16}$$

After this, we update the center points by calculating the mean value of all data points present in the cluster. The new center points are selected so as to minimize the dissimilarity within the newly formed clusters. This can be quantified as:

$$S_d = \sum_{i=1}^{k} \sum_{x \in C_i} (C_i - x)^2 \tag{17}$$

Where S_d represents the sum of squared distances from each point to its respective center.

Finally, steps above are repeated in a cycle to create better, more precise clusters through iteration.

The k-means algorithm finds application in various domains like image analysis, and document processing. However, its performance significantly relies on the initial selection of centers. Inappropriate initial center choices could lead to imperfect clustering. To overcome this, it is advisable to repeat the initialization of k-means using different sets of centers. An alternative initialization strategy, called k-means++, was proposed [7]. In this approach, centers are chosen randomly with the probability being proportional to the squared distance from already chosen centers.

While k-means is useful in data compression and artificial intelligence, it suffers from a high computational load, which results in increased time cost.

2.2.2. Mean-shift The mean-shift algorithm is a non-parametric technique used to identify clusters in a dataset. It iteratively shifts data points towards the mean of points within a specified radius until convergence is reached [8].

The general formula is:

$$m(a) = \frac{\sum_{a_j \in N(a)} K\left(a_j - a\right) a_j}{\sum_{a_j \in N(a)} K\left(a_j - a\right)} \tag{18}$$

where $N(a)$ refers the neighborhood of a, a set of points for which $K(a_j - a) \neq 0$

To initiate the mean-shift algorithm, a kernel function $K\left(x_i - x\right)$ is chosen with an initial guess x, which assigns weights to each point for the calculation of the mean shift vector. A Gaussian function is often chosen as the kernel function, and its bandwidth parameter determines the radius size used in the mean shift computation.

After this, an initial guess for the cluster centers is made by selecting a subset of data points as starting points. For every starting point, the mean shift vector is calculated by obtaining the weighted mean of data points within the radius defined by the kernel function. This starting point is then shifted towards the mean shift vector, and the process is repeated until convergence.

Convergence is generally considered when the mean shift vector falls below a particular threshold. At this point, the starting point is recognized as a cluster center, and all points within a certain distance of the cluster center are assigned to that cluster.

The mean-shift algorithm's performance heavily relies on the selection of the kernel function and the bandwidth parameter. A large bandwidth leads to larger clusters, while the choice of the kernel function could affect the algorithm's efficacy for different data types.

DOI: 10.54254/2755-2721/31/20230153

The advantage of mean-shift is its ability to automatically determine the number of clusters based on the data. It is also robust to outliers as the mean shift vector is computed based on the weighted mean of nearby points rather than the precise location of each point.

However, mean-shift can be computationally demanding for large datasets, as the distance between each pair of data points needs to be calculated multiple times during the iterative process. It is also sensitive to the initial starting points as the algorithm may converge to local optima rather than the global optimum.

Mean-shift is widely used in computer vision, image processing, and the analysis of data. By identifying clusters in the data, it uncovers underlying patterns and relationships that aid in further analysis and decision-making.

2.2.3. DBSCAN DBSCAN forms clusters according to density measurements, as clusters typically exhibit higher density than the remaining parts of the dataset. Density in the data space is measured to discern this [9]. A cluster in this method is seen as the maximum set of density-connected points. Before the process begins, three factors need to be determined: D, a dataset with n objects; R, the radius parameter; and $MinP$, the neighborhood density, defined as the minimum number of points that are within a specified radius of each other [10].

The process of finding clusters involves four steps:

Initially, all neighboring points within Eps are identified and the core points, which have more than $MinP$ neighbors, are established as per:

$$N_{eps}(q) : p \in D | dist(p, q) \leq R \tag{19}$$

Furthermore, core points must have at least $MinP$ within R from themselves as per:

$$|N_{eps}(p)| \geq MinP \tag{20}$$

Subsequently, for each core point that is not part of a cluster, a new cluster is created.

Following this, all neighbor points will be determined recursively. They will be assigned to the nearest cluster.

Finally, the steps above are repeated until visiting all of the core points. The points which do not belong to any clusters are considered noise.

DBSCAN is particularly beneficial for dense data and provides better results than the k-means algorithm. It also identifies outliers.

Despite its wide application in many fields with the advancement of technology, DBSCAN has its limitations. For high-dimensional data, defining density is more challenging, and computational overhead can be significant as all nearest neighbor points need to be calculated.

2.2.4. Spectral Clustering In the context of machine learning, spectral clustering is a powerful technique that uses the eigenvalues of a matrix to perform dimensionality reduction before applying clustering methods. This method employs the mathematically elegant framework of graph theory, providing effective results in numerous scenarios. We will discuss some fundamental concepts of spectral clustering: the undirected weight map, the similarity matrix, the Laplacian matrix, and the undirected graph cut. We will then walk through the overall process of spectral clustering.

The Undirected Weight Map, a crucial component of spectral clustering, serves as a weighted adjacency matrix, illustrating the weights of edges connecting pairs of nodes in an undirected graph. If no link exists between nodes, the corresponding element in the map is typically zero[11]. Formally, we can express this as:

$$W(i, j) = \begin{cases} \text{weight of the link between nodes } i \text{ and } j & \text{if an edge is present} \\ 0 & \text{otherwise} \end{cases} \tag{21}$$

DOI: 10.54254/2755-2721/31/20230153

Meanwhile, the Similarity Matrix is a symmetric structure encapsulating the resemblance between pairs of data points, an essential factor in various machine learning algorithms. The matrix elements represent the similarity between point pairs, with a higher value indicating a stronger likeness[12]. The formula for similarity is:

$$S_{ij} = \text{degree of likeness between points } i \text{ and } j \tag{22}$$

The Laplacian Matrix, frequently referred to as the graph Laplacian, provides a matrix representation of a graph. We define it as the difference between the graph's degree matrix and the adjacency matrix [8], denoted as:

$$L = D - A \tag{23}$$

Here, D represents the degree matrix — a diagonal matrix with node degrees on the diagonal, and A is the adjacency matrix of the graph.

The concept of the Undirected Graph Cut measures the total weight of edges required to partition the graph into disjoint subsets[13]. We can mathematically express it as:

$$\text{cut}(S, T) = \sum_{i \in S} \sum_{j \in T} W(i, j) \tag{24}$$

In this equation, S and T are disjoint node subsets of the graph, and W_{ij} is the weight of the edge linking nodes i and j.

With the fundamental concepts in place, let's explore the procedure of spectral clustering:

Constructing Similarity Graph: The first step is to construct a similarity graph given a set of data points $x_1, x_2, ..., x_n$. In this graph, each data point is represented as a node, and the weight of the edges signifies the similarity between two data points. Common similarity measures include the ϵ-neighborhoods, the Gaussian (heat) kernel, and the k-nearest neighbors.

Degree matrix and Laplacian computation: The process continues with the computation of the degree matrix D and the Laplacian L. The degree matrix in an undirected graph is a diagonal matrix that encapsulates the degree — the number of edges attached to each vertex. The Laplacian matrix is calculated as $L = D - W$, with W being the adjacency matrix of the graph.

Eigenvalues and Eigenvectors: This step involves calculating the eigenvalues and corresponding eigenvectors of the Laplacian matrix. The eigenvectors that correspond to the smallest eigenvalues are utilized[14].

New Data Matrix Formation: A matrix Y is formed from the k eigenvectors corresponding to the smallest k eigenvalues.

Row Normalization: Each row in matrix Y is normalized to have a unit length, meaning that $||y_i|| = 1$.

Clustering: Each row of Y is treated as a point in $\mathbb{R}^k$, and these points are subsequently clustered using a method such as k-means.

Spectral clustering, through its employment of graph theory and linear algebra, often outperforms traditional clustering techniques, especially in situations where the cluster shape is complex or the data is high-dimensional. Therefore, understanding and implementing spectral clustering is crucial for machine learning practitioners.

3. Experiments

3.1. Experiment Datasets and Evaluation Measures

Our experiment leveraged the widely recognized Breast Cancer dataset, a traditional binary classification dataset found in the sklearn.datasets module. The dataset comprises 30 attributes and a target variable indicating the breast cancer type ('malignant' or 'benign'). The attributes are derived from a digitized fine needle aspirate (FNA) image of a breast mass, providing detailed information on the cell nuclei in the image.

DOI: 10.54254/2755-2721/31/20230153

For assessment of the efficacy of the dimensionality reduction techniques, we used visual inspection of the scatter plots generated from the data with reduced dimensions, in addition to a comparison of the dataset's dimensionality before and after the reduction.

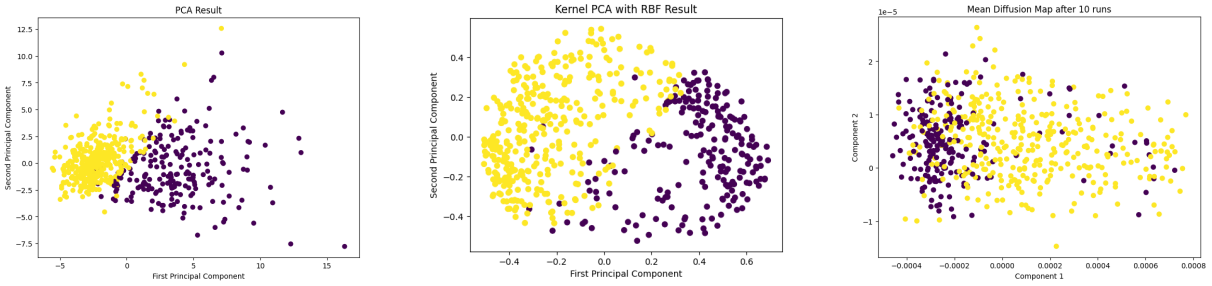
3.2. Outcomes of Dimensionality Reduction

This segment elucidates the outcomes of applying dimensionality reduction methodologies to the Breast Cancer dataset. The three approaches used were Principal Component Analysis (PCA), Kernel Principal Component Analysis (kPCA), and Diffusion Map.

3.2.1. Principal Component Analysis (PCA) PCA was performed on the standardized Breast Cancer dataset with the intention of maintaining 99

3.2.2. Kernel Principal Component Analysis (KPCA) We used KPCA with an RBF kernel to identify non-linear correlations in the dataset. By adjusting the component number to two, we were able to represent the dataset in a two-dimensional feature space. The scatter plot showing the modified data is shown in Figure 1b.

3.2.3. Diffusion Map Lastly, we applied the Diffusion Map approach to the dataset. The parameters were configured to repeat the algorithm 10 times and calculate the average outcome. As evident in Figure 1c, the purple color-coded points appear relatively grouped together, albeit with some degree of scattering. The yellow points, conversely, exhibit a high level of dispersion, hinting at a possibly complex structure in the dataset that may not be entirely captured by linear or kernel-based methodologies[2, 1].



(a) PCA scatter plot (b) kPCA with RBF scatter plot (c) Diffusion Map scatter plot

Figure 1: Dimensionality reduction results using PCA, kPCA, and Diffusion Map.

3.3. Clustering Results

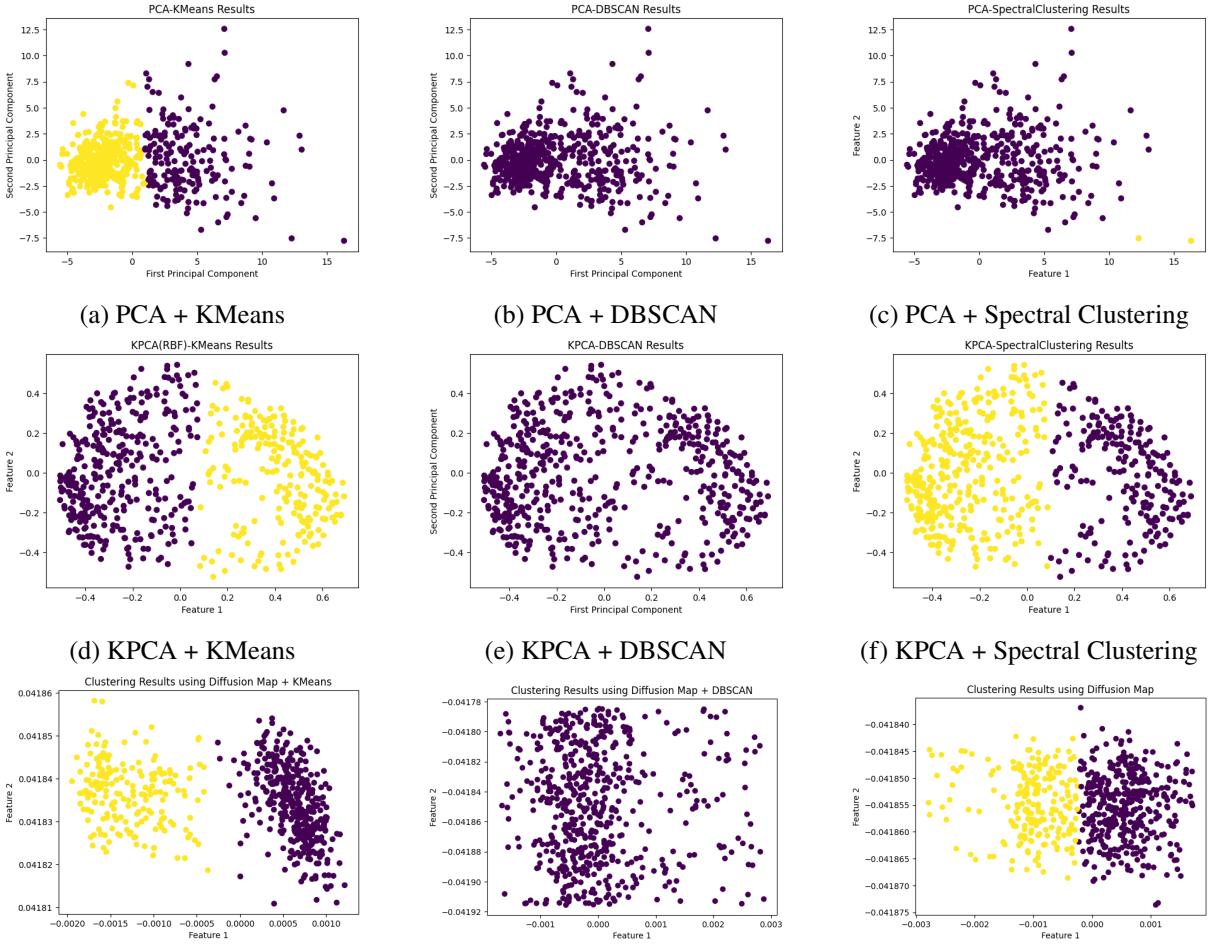
In this section, we apply dimensionality reduction techniques to the Breast Cancer dataset and summarize the obtained results. The scatterplot in figure1a shows the data structure captured by PCA in the two-dimensional principal component space, and the distribution of data points is reasonable. In contrast, Figure 1b shows the scatter plot obtained by kPCA using the RBF kernel function. It is worth noting that the distribution of data points in the results of kPCA is more scattered, which may bring certain challenges in the subsequent cluster analysis, because the degree of overlap between data points increases and the degree of discrimination between different categories decreases. In summary, our analysis results show that PCA can effectively capture the underlying structure of the data set and provide a good basis for clustering tasks; while the results obtained by kPCA using the RBF kernel function show that the data points are more scattered. features.

DOI: 10.54254/2755-2721/31/20230153

3.3.1. K-Means Clustering K-Means was applied to the PCA and kPCA transformed data. For both transformations, the K-Means algorithm showed good results with clear separations between the clusters. The results are shown in Figures 2a and 2d respectively.

3.3.2. DBSCAN Clustering DBSCAN was also applied to both the PCA and kPCA transformed data. However, the results were not satisfactory. In both cases, the clusters were not well separated and all the data points ended up in the same cluster. The results are shown in Figures 2b and 2e respectively.

3.3.3. Spectral Clustering Lastly, Spectral Clustering was applied. For the PCA transformed data, the results were not good. However, when applied to the kPCA transformed data, the Spectral Clustering algorithm performed well with clear separations between the clusters. The results are shown in Figures 2c and 2f respectively.



(a) PCA + KMeans (b) PCA + DBSCAN (c) PCA + Spectral Clustering

(d) KPCA + KMeans (e) KPCA + DBSCAN (f) KPCA + Spectral Clustering

(g) Diffusion + Kmeans (h) Diffusion + DBSCAN (i) Diffusion + SpectralClustering

Figure 2: Clustering results using PCA and kPCA with KMeans, DBSCAN, and Spectral Clustering algorithms.

3.3.4. Comparative Analysis of Dimensionality Reduction and Clustering Combinations The performance of different combinations of dimensionality reduction and clustering techniques on the dataset varied. PCA, when paired with KMeans, yielded good results with clear separation of the two

DOI: 10.54254/2755-2721/31/20230153

main components (Figure 2a), but it did not perform as well when paired with DBSCAN or Spectral Clustering, which resulted in plots with insufficient cluster separation (Figures 2b and 2c). In contrast, KPCA showed consistent performance when paired with both KMeans and Spectral Clustering, resulting in clear separation of the two main components (Figures 2d and 2f). However, similar to PCA, KPCA did not yield satisfactory results when paired with DBSCAN, leading to a lack of clear clusters (Figure 2e).

3.4. Analysis of Spectral Clustering results:

By comparing 2- (c) , 2- (f) , 2- (i) , we can conclude that pca and spectrl clustring do not work well together for breast cancer data, and the two non-linear dimensionality reduction are more effective than linear dimensionality reduction, where kpca is more effective after dimensionality reduction using spectral clustring.

3.5. Analysis of diffusion map results:

By comparing 2- (g) , 2- (h) , 2- (i) , it can be concluded that diffsuion and kmeans work best together for breast cancer data.The results of diffsuion and spectral clustring pairing are unstable.

3.6. Evaluation of clustering results

Purity Results for breast cancer datataset:

	PCA	KPCA	DiffusionMaps
DBSCAN	0.862917	0.627417	0.627417
KMeans	0.906854	0.905097	0.699649
SpectralClustering	0.903339	0.901582	0.668366

ARI Results for breast cancer datataset:

	PCA	KPCA	DiffusionMaps
DBSCAN	0.464049	0.000000	0.000000
KMeans	0.659231	0.654857	0.161859
SpectralClustering	0.648825	0.643464	0.101467

Figure 3: Evaluation of clustering results for breast cancer

The breast cancer dataset is a common set used for classification tasks. We employed three different dimensionality reduction methods: PCA (a linear method), KPCA (a non-linear method), and Diffusion Map (a non-linear method), and applied three different clustering algorithms: DBSCAN, K-Means, and Spectral Clustering.

Firstly, from the perspective of Purity metric, both PCA and KPCA scored highly (¿ 0.9) when paired with K-Means and Spectral Clustering. This suggests that these two reduction methods could capture both linear and non-linear structures of the breast cancer dataset effectively, and K-Means and Spectral Clustering could perform well in clustering on the reduced data. However, the Diffusion Map performed poorly across all clustering methods, which may indicate that this reduction method didn't fully capture the intrinsic structure of the data, or further adjustments to its parameters may be needed.

Looking at ARI (Adjusted Rand Index), a measure of the consistency between clustering results and true labels, again PCA and KPCA exhibited higher ARI scores when paired with K-Means and Spectral

DOI: 10.54254/2755-2721/31/20230153

Clustering, indicating a high consistency between clustering results and actual labels. However, the results from Diffusion Map were considerably poorer across all clustering methods.

In summary, PCA and KPCA demonstrated to be the more effective dimensionality reduction methods for this breast cancer dataset, while DBSCAN generally performed poorly. This could be due to DBSCAN's sensitivity to noise and the shape of clusters, while the breast cancer dataset may contain significant noise or its cluster shapes may not be suitable for DBSCAN.

4. Conclusion

High-dimensional data can lead to computational challenges and uneven data distribution due to reduced data point density. Therefore, reducing dimensionality aids in better understanding and data exploration. Clustering methods prove effective for reducing data dimensionality. In this paper, we review several dimensionality reduction and clustering techniques,We also explore Spectral Clustering, a technique that combines clustering and dimension reduction concepts. According to the clustering results, we evaluate the efficiency of these techniques empirically.

For future research, on the one hand, we could further adjust parameters of Diffusion Map to see if performance could be improved. On the other hand, we could also try more non-linear dimensionality reduction methods such as Isomap or t-SNE to see how they perform on this dataset. Additionally, more clustering algorithms like hierarchical clustering or density peak clustering could be employed for more comprehensive experimental results.

Acknowledgements

JunFan Liu,Weide He contributed equally to this work and should be considered co-first authors.

References

[1] Ronald R Coifman and Stéphane Lafon. Diffusion maps. *Applied and computational harmonic analysis*, 21(1):5–30, 2006.

[2] Per-Erik Danielsson. Euclidean distance mapping. *Computer Graphics and image processing*, 14(3):227–248, 1980.

[3] Karl J Friston. Functional and effective connectivity: a review. *Brain connectivity*, 1(1):13–36, 2011.

[4] Haibin Ling and Kazunori Okada. Diffusion distance for histogram comparison. In *2006 IEEE Computer Society Conference on Computer Vision and Pattern Recognition (CVPR'06)*, volume 1, pages 246–253. IEEE, 2006.

[5] Mohiuddin Ahmed, Raihan Seraj, and Syed Mohammed Shamsul Islam. The k-means algorithm: A comprehensive survey and performance evaluation. *Electronics*, 9(8):1295, 2020.

[6] K Krishna and M Narasimha Murty. Genetic k-means algorithm. *IEEE Transactions on Systems, Man, and Cybernetics, Part B (Cybernetics)*, 29(3):433–439, 1999.

[7] David Arthur and Sergei Vassilvitskii. How slow is the k-means method? In *Proceedings of the twenty-second annual symposium on Computational geometry*, pages 144–153, 2006.

[8] Miguel A Carreira-Perpinán. A review of mean-shift algorithms for clustering. *arXiv preprint arXiv:1503.00687*, 2015.

[9] Michael Hahsler, Matthew Piekenbrock, and Derek Doran. dbscan: Fast density-based clustering with r. *Journal of Statistical Software*, 91:1–30, 2019.

[10] Kamran Khan, Saif Ur Rehman, Kamran Aziz, Simon Fong, and Sababady Sarasvady. Dbscan: Past, present and future. In *The fifth international conference on the applications of digital information and web technologies (ICADIWT 2014)*, pages 232–238. IEEE, 2014.

[11] Andrew Ng, Michael Jordan, and Yair Weiss. On spectral clustering: Analysis and an algorithm. *Advances in neural information processing systems*, 14, 2001.

[12] Seyoung Park and Hongyu Zhao. Spectral clustering based on learning similarity matrix. *Bioinformatics*, 34(12):2069–2076, 2018.

[13] Marco Di Summa, Andrea Grosso, and Marco Locatelli. Branch and cut algorithms for detecting critical nodes in undirected graphs. *Computational Optimization and Applications*, 53:649–680, 2012.

[14] A Cantoni and P Butler. Eigenvalues and eigenvectors of symmetric centrosymmetric matrices. *Linear Algebra and its Applications*, 13(3):275–288, 1976.