Implementing t-SNE for Various Datasets

1st Ajay Gangwar
Department of Computer Science
IIT Ropar

2nd Subodh Bijwe
Department of Computer Science
IIT Ropar

***Abstract*—There are patterns embedded in the data and because of the high dimensionality of data, those patterns cannot be easily detected by visual inspection. This is the key reason why high-dimensional data visualization is relevant. Dimensionality Reduction is necessary to accomplish this visualization objective. However no technique was able to maintain the local data structure (i.e. in High Dimensions) until 2008 and also visualize the data. A new technique, 't-SNE', was developed by Laurens van der Maaten and Geoffrey Hinton in 2008, which is capable of capturing most of the local data structure in high-dimensional while also visualizing it in 2D. In this paper, we tried to explore t-SNE and compare it with another similar techniques.**

***Index Terms*—Isomap, Embedding, Neighbourhood, Stochastic, divergence**

I. Introduction

Visualization of data has always been an important issue for data scientists and data engineers. IBM has estimated that every day around 2.5 quintillion bytes of data is created - so much that almost 90% of data that is there in the world today is generated in the past few years. This data includes images, videos, audio files and texts; some company specific data like what you bought, time of the day those things are bought, flight bookings, doctor's appointment, criminal lawsuits, news headlines, etc.

With the advances of machine learning and pattern recognition techniques it gets easier to find recurring patterns in data. Even so, data scientists need to visualise how the data looks in order to know which method to use for different tasks like classification or clustering. Over the years multiple dimensionality reduction techniques as well as visualization techniques have been proposed and are currently used in industry as well as academia. In this paper, we survey some of them and then focus on implementing t-SNE, which performs significantly better than all other techniques.

II. Literature Survey

For introduction we studied different dimensionality reduction and visualization techniques on the Swiss Roll dataset (Figure 1) and the results are shown in Figure 4. A small explanation with implementation of each technique will be given in this section.

A. Principal Component Analysis (PCA)

PCA is used in exploratory data analysis and for making predictive models. It is commonly used for dimensionality reduction by projecting each data point onto only the first few principal components to obtain lower-dimensional data while preserving as much of the data's variation as possible. The first principal component can equivalently be defined as a direction that maximizes the variance of the projected data. The i^{th} principal component can be taken as a direction orthogonal to the first $i-1$ principal components that maximizes the variance of the projected data.

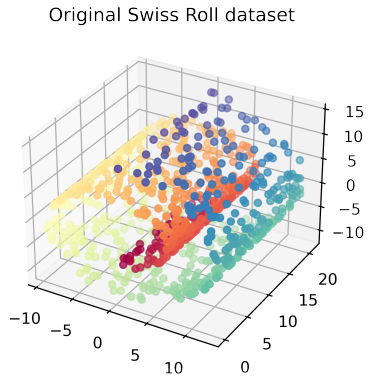

Fig. 1. Original Swiss Roll Dataset

B. Isomap

A high level description of Isomap can be summarized as below,

- Find the points lying in the neighbourhood of each point (can be done using K nearest neighbours or by finding all points in a given radius)
- Construct a neighbourhood graph (a graph in which close points are connected via an edge)
- Compute shortest path between two nodes (using Dijkstra's or Floyd-Warshall's algorithm)
- Compute low dimensional embedding

C. Locally Linear Embedding (LLE)

The basic idea of locally linear embedding is to take a high dimensional manifold and cast it into a low dimensional space while preserving the geometric features of the manifold. LLE takes advantage of local geometry and pieces it together to preserve the global geometry on a lower dimensional space.

III. METHODOLOGY

The content is taken from [1] and [2].

A. *Stochastic Neighbor Embedding(SNE)*

SNE begins by embedding higher dimensional Euclidean distances between the datapoints into conditional probabilities which are then used to represent similarities between those datapoints. If there is a Gaussian centered at datapoint x_i then, $p_{j|i}$,the similarity between datapoint x_i and x_j, is the conditional probability that x_i would select x_j as it's neighbour, if the neighbours were supposed to be picked in proportion to their PDF. So, for datapoints that are closer to each other, $p_{j|i}$, is comparatively high than for the datapoints that are at farther distances, almost negligible. $p_{j|i}$ is defined as,

$$p_{j|i} = \frac{exp(-||x_i - x_j||^2/2\sigma_i^2)}{\sum_{k\neq i} exp(-||x_i - x_k||^2/2\sigma_i^2)}, \tag{1}$$

where σ_i is the variance of the Gaussian centered at x_i. Values of $p_{i|i}$ is set to zero because only pairwise modelling is done.

For low dimensional representations, y_i and y_j, of datapoints x_i and x_j a similar conditional probability is calculated, which is denoted by $q_{j|i}$. The variance of the Gaussian at y_i is now set to $\frac{1}{\sqrt{2}}$. Hence, the mapping takes the form,

$$q_{j|i} = \frac{exp(-||y_i - y_j||^2)}{\sum_{k\neq i} exp(-||y_i - y_k||^2)}. \tag{2}$$

Here as well, $q_{i|i} = 0$

If the low dimensional datapoints y_i and y_j correctly models the higher dimensional datapoints x_i and x_j, then the conditional probabilities $p_{j|i}$ and $q_{j|i}$ will be equal. This gives us a minimization problem where we try to minimize the mismatch between $p_{j|i}$ and $q_{j|i}$. One of the best measure to estimate differences between probabilities is the Kullback-Leibler divergence (or the KL divergence). In SNE, the sum of KL divergences over all datapoints are minimized using gradient descent. The cost function C is defined as,

$$C = \sum_i KL(P_i||Q_i) = \sum_i \sum_j p_{j|i} \log \frac{p_{j|i}}{q_{j|i}}, \tag{3}$$

Here, P_i is the conditional probability distribution over all other datapoints when datapoint x_i is given. And Q_i is the conditional probability distribution over all other mapped-datapoints when mapped-datapoint y_i is given.

The cost function of SNE focuses on retaining the local structure of the data in lower dimension. In particular, a huge cost is incurred if small $q_{j|i}$ is used to model large $p_{j|i}$ but a small cost for vice versa. This is because farther points result in smaller $p_{j|i}$ and closer points result in greater $p_{j|i}$ values.

For the minimization the calculated gradient as the form,

$$\frac{dC}{dy_i} = 2\sum_j (p_{j|i} - q_{j|i} + p_{i|j} - p_{i|j})(y_i - y_j) \tag{4}$$

To choose appropriate variance of the centered Gaussian, a binary search is performed that results in distribution entropy over the neighbours equal to $log(k)$, where k is the perplexity or suitable count of neighbours present locally. The perplexity is defined as,

$$k = 2^{H(P_i)}, \tag{5}$$

where $H(P_i)$ is the Shannon Entropy of P_i measured in bits:

$$H(P_i) = -\sum_j p_{j|i} \log_2 p_{j|i}. \tag{6}$$

Thus, for sparse data, smaller perplexity k should be chosen, which results in greater variance and neighbourhood size. Now, since Gaussian Kernel is used, probability of x_j being a neighbour of x_i decreases sharply if the point lies outside the neighbourhood of x_i, where the neighbourhood is defined by the variance of the Gaussian.

B. *Crowding Problem*

At one end where SNE preserves the local structure of data, it also faces the "crowding problem". The area in local dimension won't be enough to accommodate the data points as they were in higher dimension.

In simple words, comparatively smaller space is there to accommodate the data in lower dimension as compared to higher dimension. Thus, distinct clusters in higher dimensional space will be pushed closer to each other in 2-D or 3-D embedding. Let's see how t-SNE solves this problem.

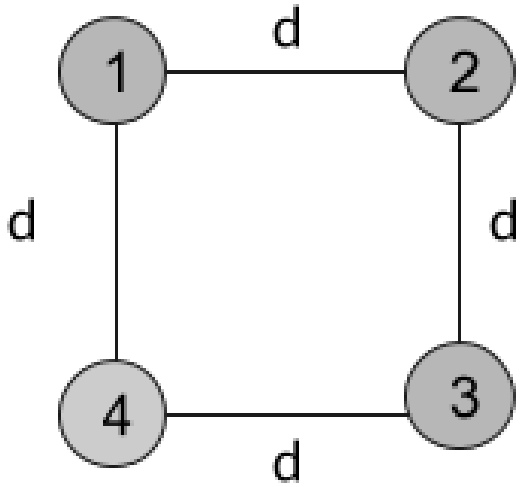

Fig. 2. Crowding Problem High Dimension

As can be seen from Figure 2, the data is in 2 dimensions and distance of each point from other is 'd'. If we try to model this in lower dimension as in 3, let's say one dimension then we will have lesser space to model. Because of that, the distances are not maintained properly. In the example

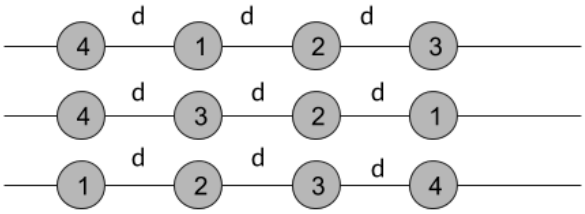

Fig. 3. Crowding Problem Low Dimension

we can see that if we try to model point '1' closer to '2' and '4' as it was in higher dimension in lower dimension, we notice that it gets impossible to model the distance from '2' to '3' or '4' to '3' as we require. This issue gets huge significance when we are working in more dimensions. Even if we consider uniformly distributed points in a 10 dimensional sphere and want to model it in 2- or 3-dimensions, we face similar problem. Due to lack of space the points tend to collide on each other or stay far apart from each other.

C. *t-Distributed Stochastic Neighbor Embedding (t-SNE)*

As opposed to SNE, t-SNE has two major differences:

- it makes SNE symmetric
- a Student's t-distribution rather than a Gaussian to computer the similarity in the low-dimensional space to avoid the crowding problem.

What does it mean to have symmetric SNE? Well, in SNE p_{ji} will not necessarily be equal to p_{ij} because σ_i won't necessarily be equal to σ_j. So, instead of choosing different σ values for different datapoints, just one σ value is chosen for all datapoints. This makes the computation easier and less prone to outliers.

Moreover, the Student's t-distribution is used instead of the Gaussian, defined as,

$$q_{ij} = \frac{(1 + ||y_i - y_j||^2)^{-1}}{\sum_{k \neq l}(1 + ||y_k - y_l||^2)^{-1}}. \tag{7}$$

and, the cost function of t-SNE is defined as,

$$C = \sum_i KL(P_i||Q_i) = \sum_i \sum_j p_{ij} \log \frac{p_{ij}}{q_{ij}}, \tag{8}$$

The gradient function takes the form,

$$\frac{dC}{dy_i} = 4 \sum_{j=1, j \neq i} (p_{ij} - q_{ij})(1 + ||y_i - y_j||^2)^{-1}(y_i - y_j) \tag{9}$$

$$= 4 \sum_{j=1, j \neq i} (p_{ij} - q_{ij}) q_{ij} Z (y_i - y_j) \tag{10}$$

$$= 4(\sum_{j \neq i} p_{ij} q_{ij} Z (y_i - y_j) - \sum_{j \neq i} q_{ij}^2 Z (y_i - y_j)) \tag{11}$$

$$= 4(F_{attraction} + F_{repulsion}) \tag{12}$$

where $Z = \sum_{l,s,l \neq s}(1 + ||y_l - y_s||^2)^{-1}$

IV. Results on different Datasets

A. Time and Space Complexities

1) PCA :- Covariance matrix computation is of $O(np \times min(n,p))$ which is a result of multiplying two matrices $(n \times p)$ and $(p \times n)$; its eigen-value decomposition is $O(p^3)$. So, the time complexity of PCA is $O(np \times min(n,p) + p^3)$ where n is the number of samples and p is the number of dimensions. The space complexity of PCA is $O(np + p^2)$.
2) t-SNE :- The algorithm computes pairwise conditional probabilities and tries to minimize the sum of the difference of the probabilities in higher and lower dimensions. So, t-SNE has a quadratic time and space complexity in the number of data points.
3) Isomap :- The overall time complexity of isomap is $O(D(logk) \times n(logn)) + O(n^2(k + logn)) + O(dn^2)$,where n is the number of datapoints,D is the input dimension,k is the number of nearest neighbours and d is the output dimension and the space complexity of isomap is $O(nd)$.
4) LLE :- The overall time complexity of standard LLE is $O(Dlogk \times nlogn) + O(Dnk^3) + O(dn^2)$, where n is the number of datapoints,D is the input dimension,k is the number of nearest neighbours and d is the output dimensionand the space complexity of LLE is $O(nd)$.

B. Datasets Used

We have chosen the datasets which are not exactly very high dimensional (except for MNIST), so the results of different algorithms can be clearly seen on it. When other algorithms simply tries to project the data into lower dimensions trying to retain maximum variance and global structure of data, t-SNE tries to preserve the local structure in it.

The results can be seen in the images below,

1) Swiss Roll Dataset (Figure 4)
2) MNIST (Figure 5)
3) Iris (Figure 6)
4) Olivetti (Figure 7)
5) Circle (Figure 8)
6) Curve (Figure 9)

C. Inferences

1) **Swiss Roll Dataset**: From 4 we noticed that t-SNE was able to model the local as well as global structure in 2-dimensions, while others just tried to preserve the global structure. We can see that LLE tried to model something that looks like a top-view of dataset while PCA and Isomap modelled it like front view of the dataset, which is a clear indication of capturing the global structure.
2) **MNIST Dataset**: MNIST is a 784 dimension dataset with 60k datapoints. From 5 we can see that t-SNE does an impressive job by finding clusters and subclusters in the data, but it is prone to getting stuck in local minima as we can see there is some overlap between the clusters.

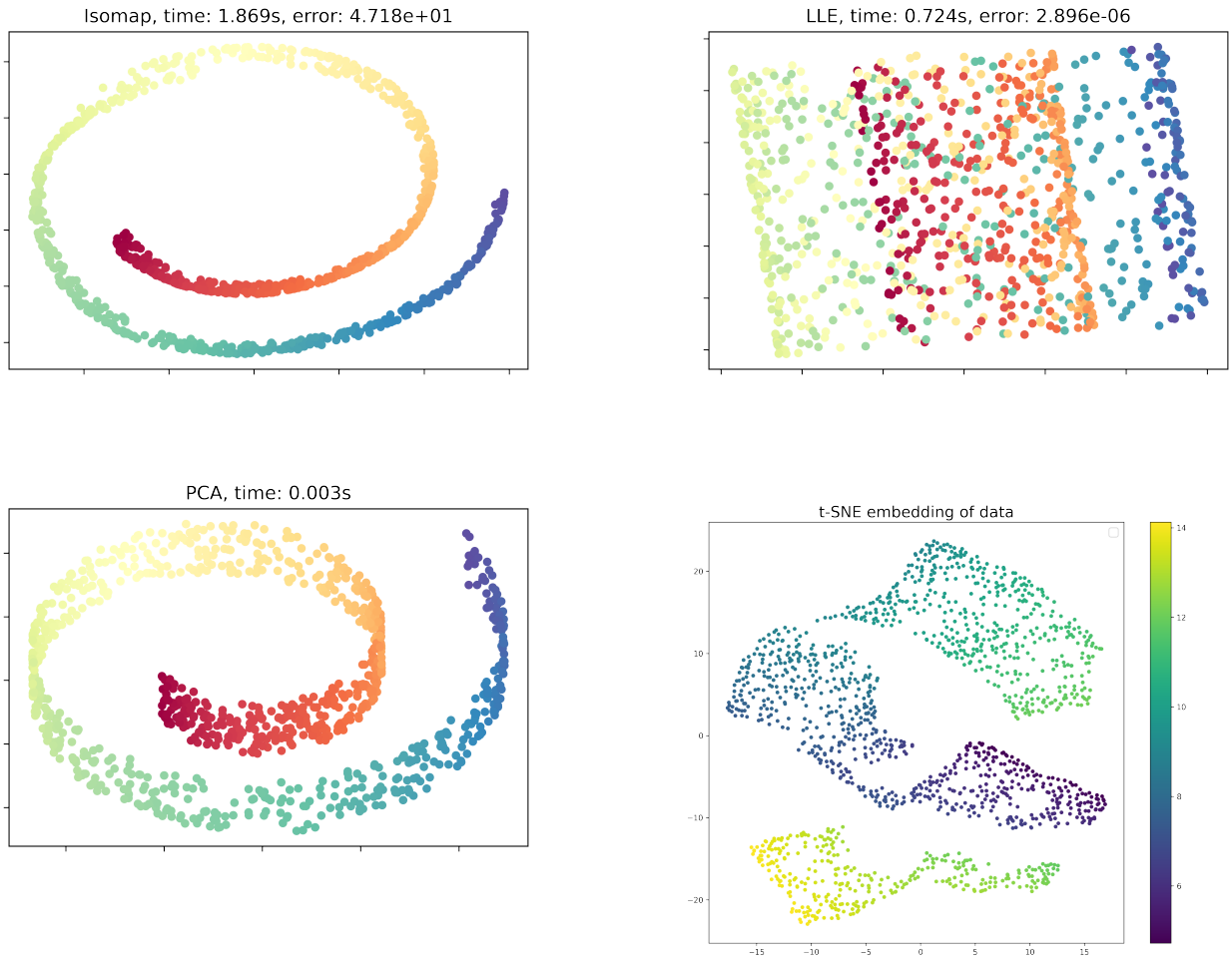

Fig. 4. Performance of various algorithms on Swiss Roll Dataset 1. Isomap(top left) 2. LLE(top right) 3. PCA(bottom left) 4. t-SNE(bottom right)

Overall the visualization achieved by tSNE is far better than other algorithms on MNIST dataset. Isomap and PCA were also able to separate some clusters very well but LLE performs worst.

3) **Iris Dataset**: It is a 4 dimension dataset with 150 points. From 6 we can see that iris-setosa is very well separated while there is a small overlap between iris-versicolor and iris-virginica in case of tSNE and we can use simple line or if-else condition to classify them. PCA and Isomap had also done a decent job in separating the flowers while LLE was not able to separate them properly.
4) **Olivetti faces Dataset**: It is a 4096 dimension dataset with 400 datapoints. From 7 we can see that t-SNE was the only algorithm which had separated the datapoints properly and no other algorithm was able to separate the datapoints.
5) **Circle Dataset**: It is a 2 dimension dataset with 300 points. All algorithms were able to separate the datapoints but the visualization achieved by tSNE on this simple dataset is far better than the other algorithms.
6) **Curve Dataset**: It is a 3 dimension dataset with 1000 points. PCA performs worst on this dataset as the overlap between the classes is more. Here tSNE while trying to preserve the local structure unfolds the manifold so that the classes are very well separated and we get very clean visualization through tSNE. Isomap and LLE was also able to separate the classes and performs decent job.

V. Conclusion

We conclude that in terms of dimensionality reduction and visualization, t-SNE performs better in most of the cases. The clusters formed after running t-SNE on the datasets are clearly separable and easier to cluster as compared to other techniques. It preserves local as well as global structure of the data and outperforms other top-notch visualization techniques currently being used in the Industry.

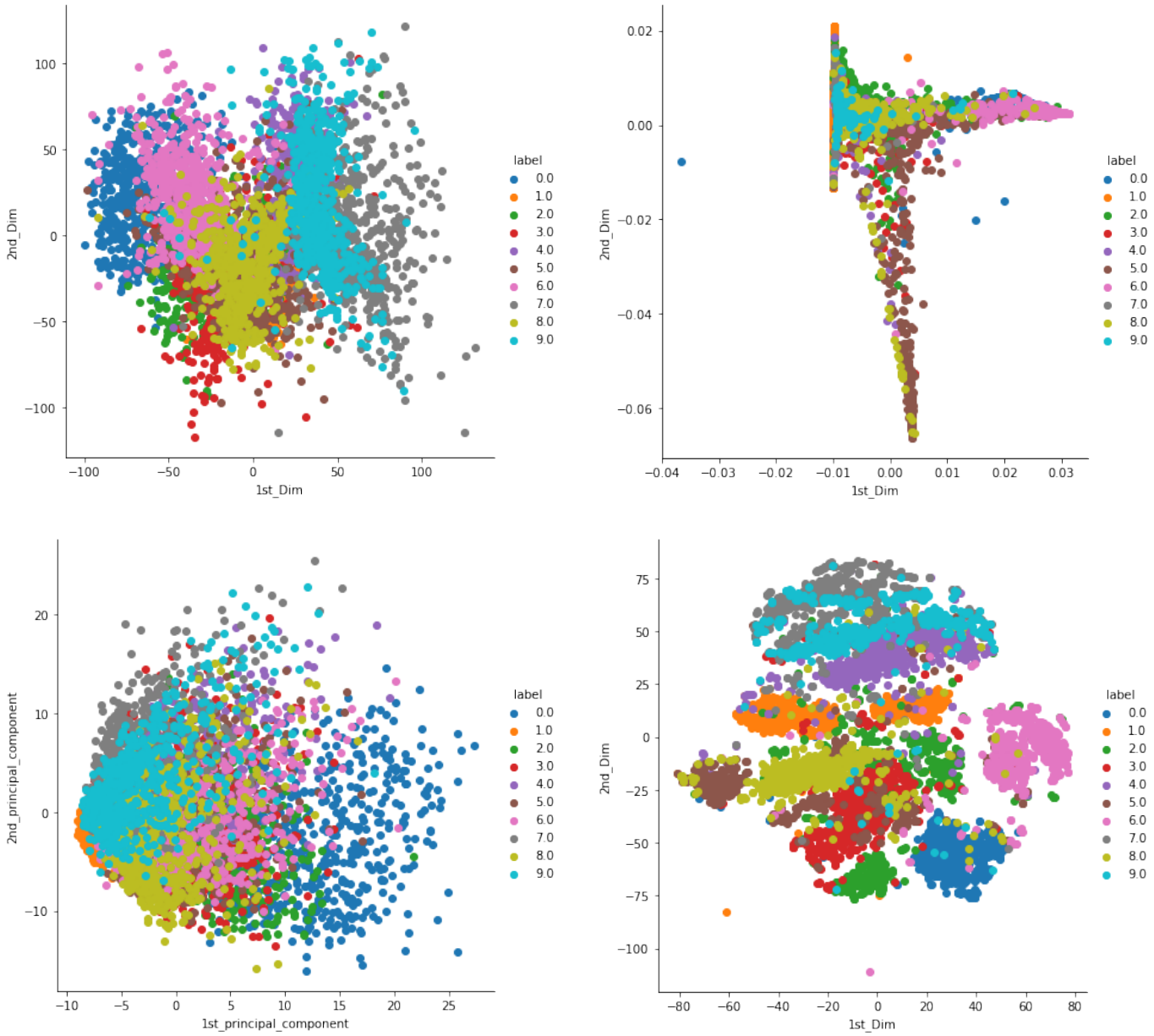

Fig. 5. Performance of various algorithms on MNIST Dataset 1. Isomap(top left) 2. LLE(top right) 3. PCA(bottom left) 4. t-SNE(bottom right)

REFERENCES

[1] Geoffrey E Hinton and Sam Roweis. Stochastic neighbor embedding. In S. Becker, S. Thrun, and K. Obermayer, editors, *Advances in Neural Information Processing Systems*, volume 15, pages 857–864. MIT Press, 2003.

[2] Laurens van der Maaten and Geoffrey Hinton. Visualizing data using t-SNE. *Journal of Machine Learning Research*, 9:2579–2605, 2008.

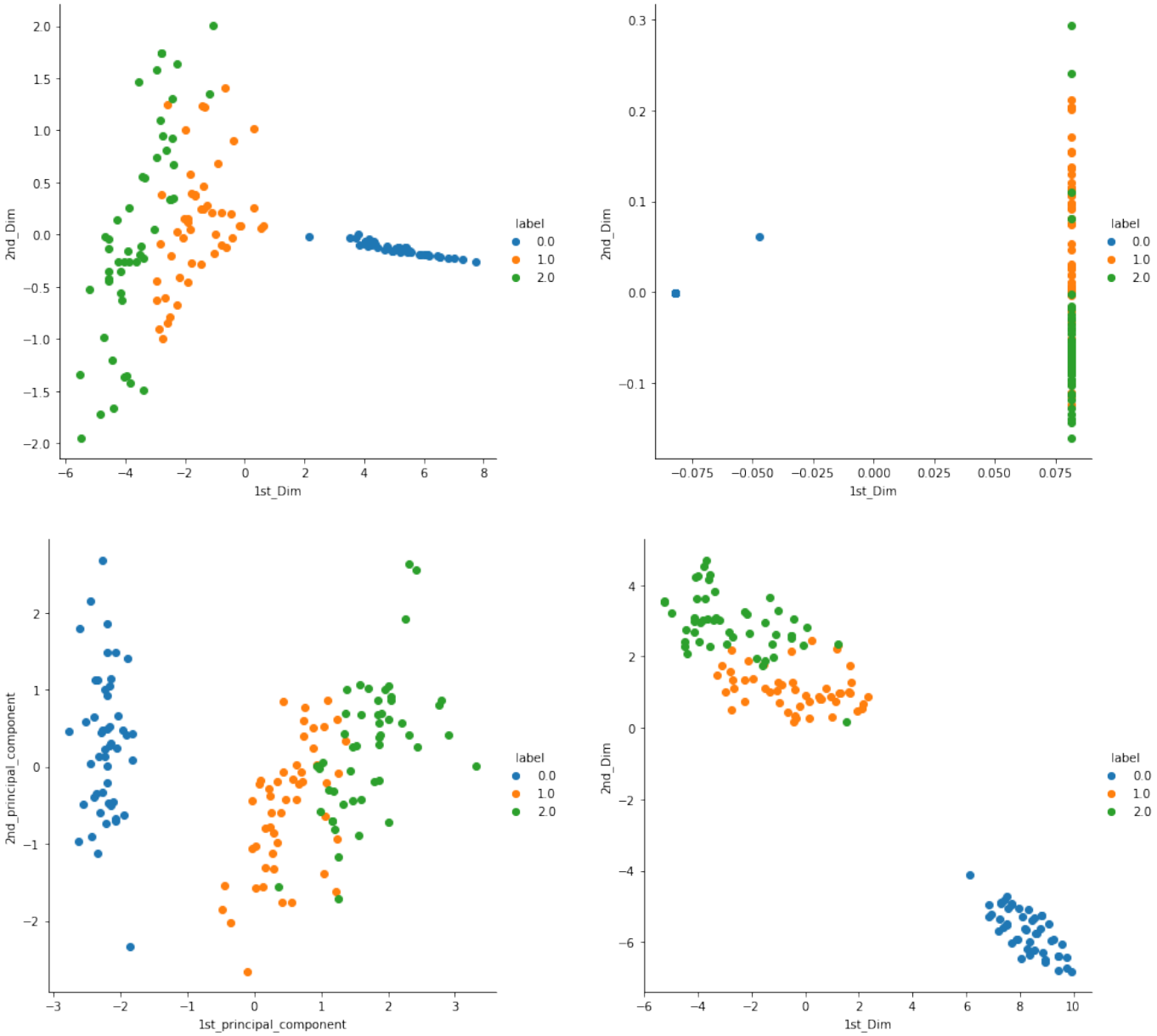

Fig. 6. Performance of various algorithms on Iris Dataset 1. Isomap(top left) 2. LLE(top right) 3. PCA(bottom left) 4. t-SNE(bottom right)

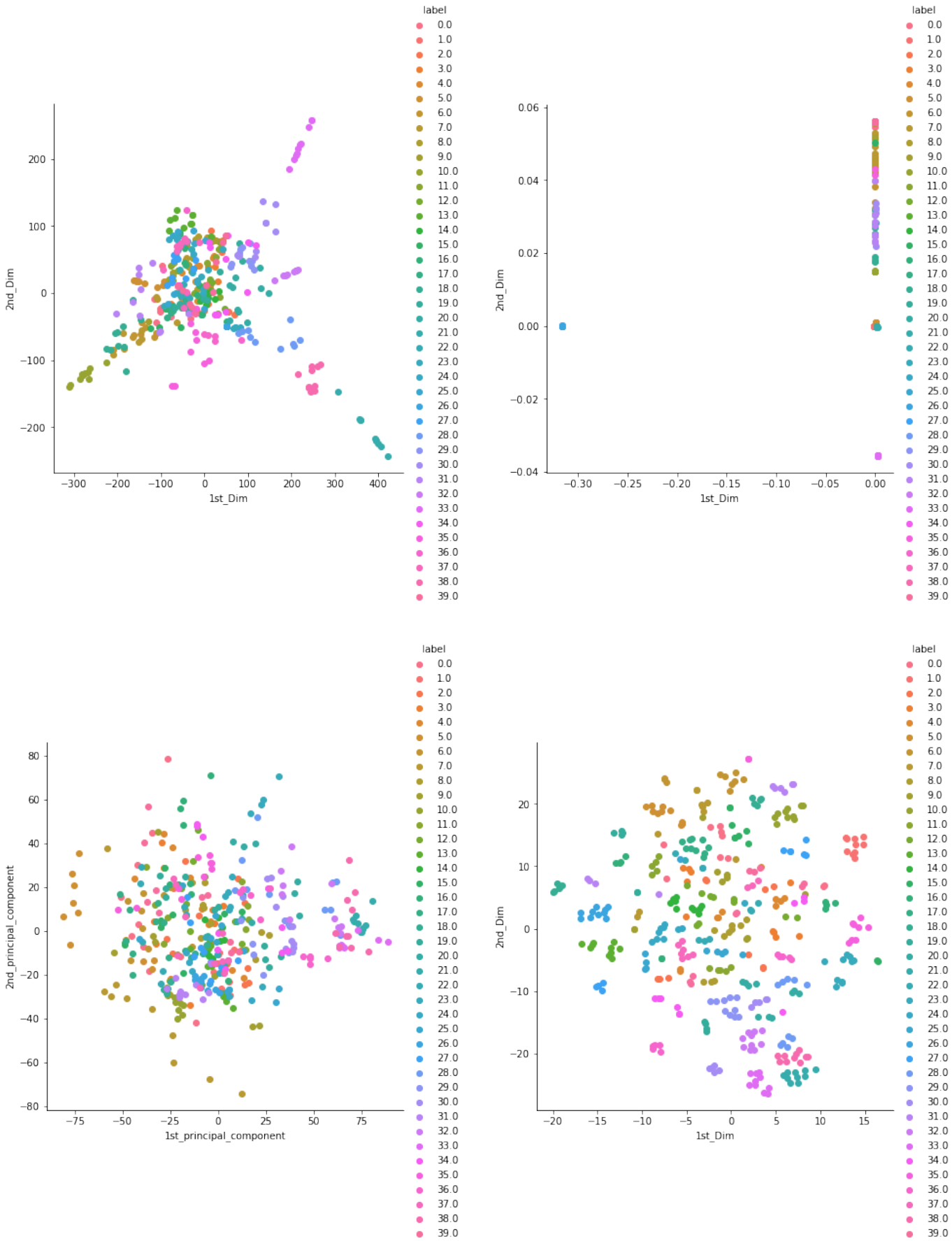

Fig. 7. Performance of various algorithms on Olivetti Dataset 1. Isomap(top left) 2. LLE(top right) 3. PCA(bottom left) 4. t-SNE(bottom right)

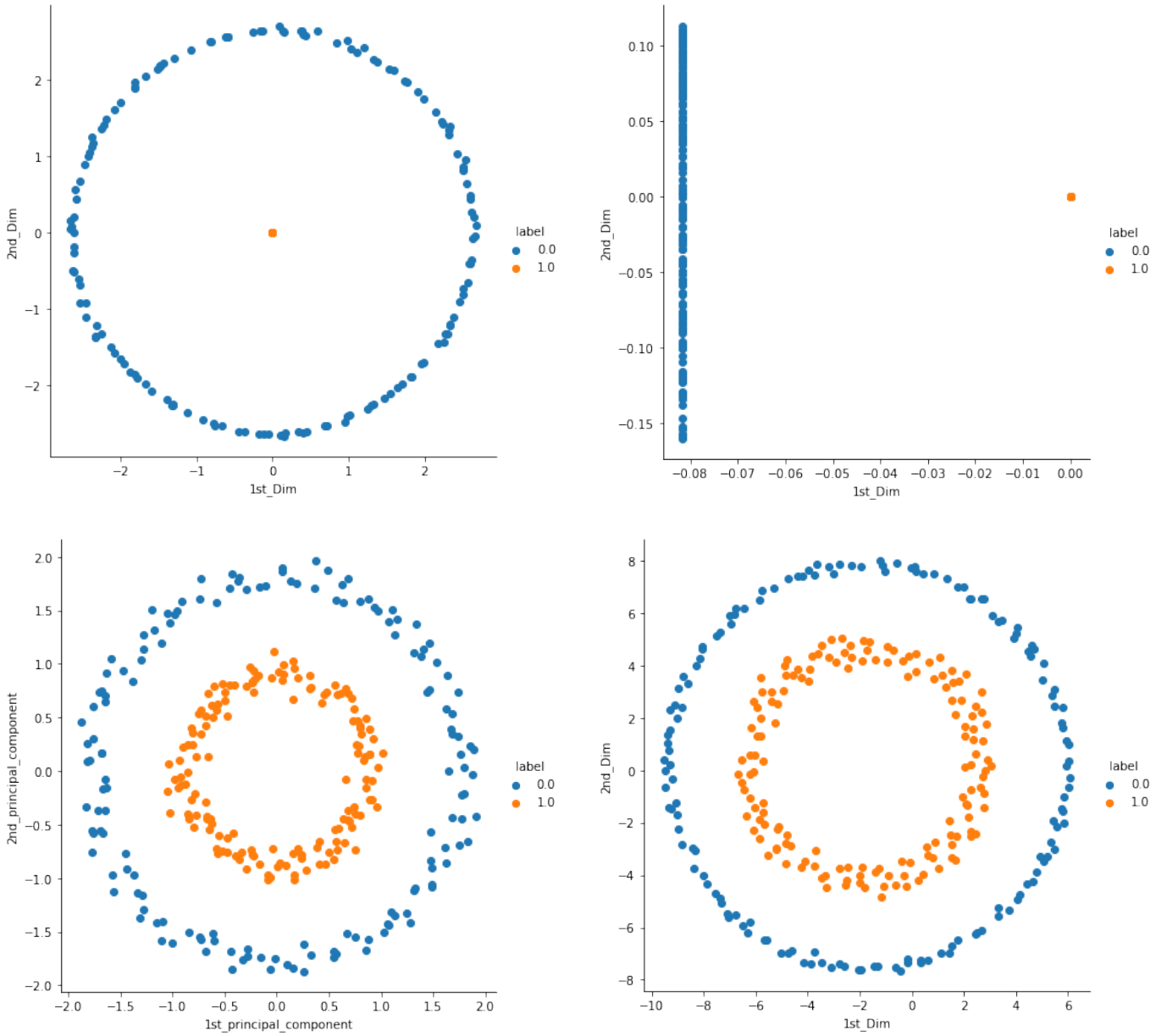

Fig. 8. Performance of various algorithms on Circle Dataset 1. Isomap(top left) 2. LLE(top right) 3. PCA(bottom left) 4. t-SNE(bottom right)

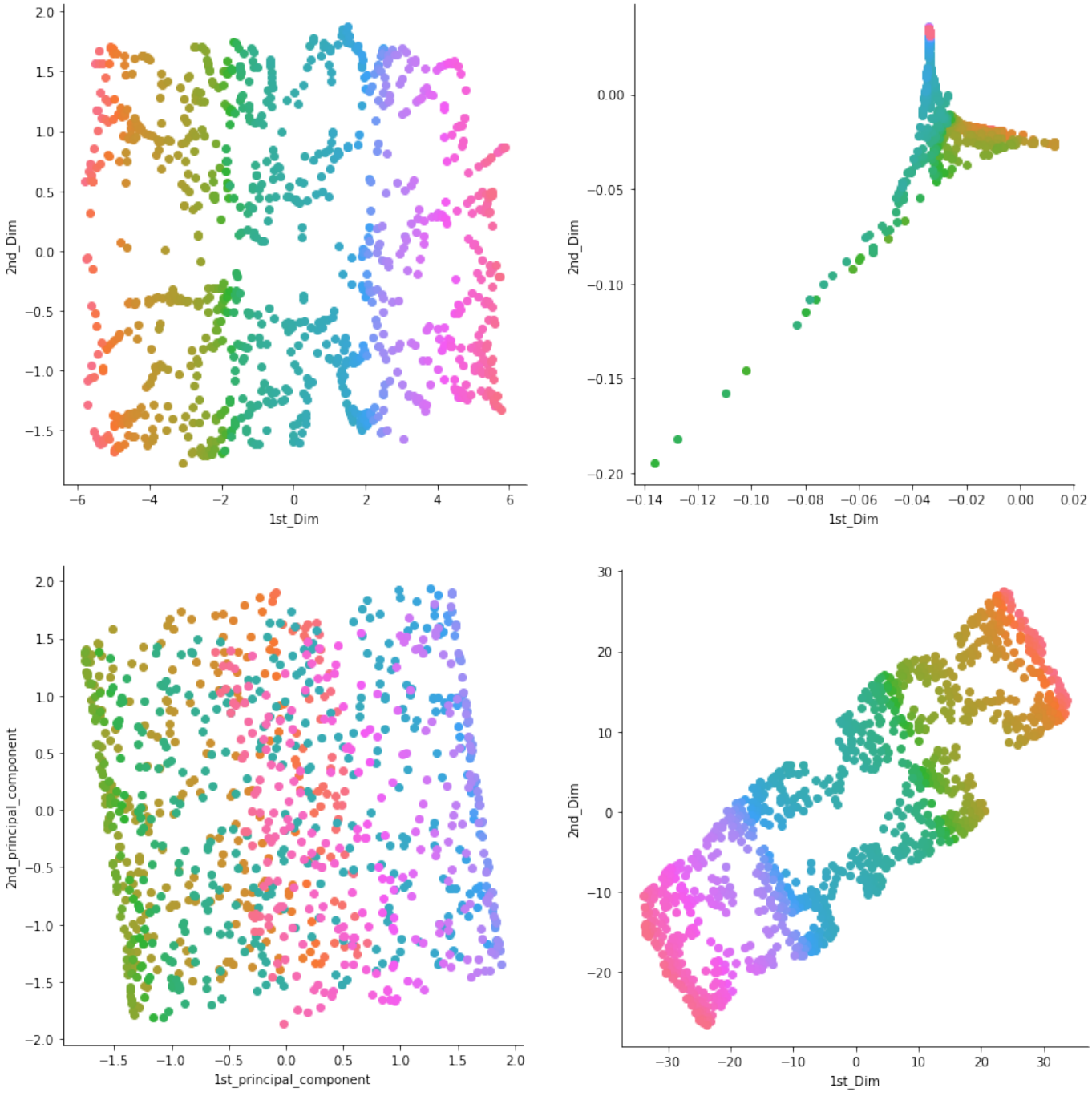

Fig. 9. Performance of various algorithms on Curve Dataset 1. Isomap(top left) 2. LLE(top right) 3. PCA(bottom left) 4. t-SNE(bottom right)