电子科技大学
University of Electronic Science and Technology of China

Subspace Learning

Feng Huang

Data Mining Lab, Big Data Research Center, UESTC
Email：junmshao@uestc.edu.cn
http://staff.uestc.edu.cn/shaojunming

Outline

数据挖掘实验室
Data Mining Lab

1.Linear Dimension Reduction

2.non-linear Dimension Reduction

3. How to detect outlier using subspace learning

Dimension Reduction

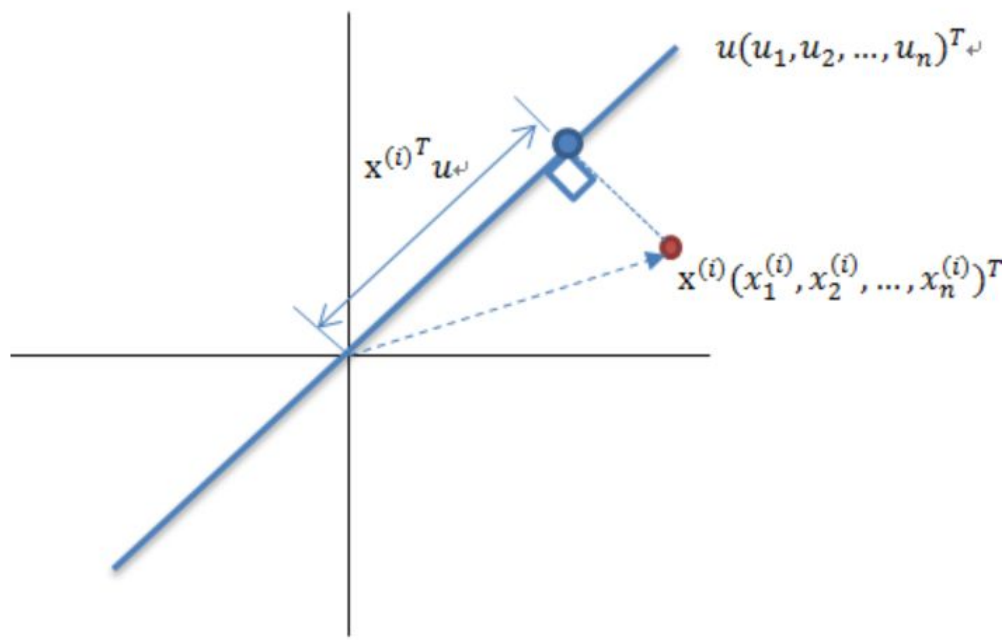
- The target of dimension reduction is to use as less basis as possible to represent raw dataset.
- The key problem of dimension reduction is what information you want to save in low-dimension representations.

PCA

Max variance Explanation

$$\frac{1}{\mathrm{m}}\sum_{\mathrm{i}=1}^{\mathrm{m}}(\mathrm{x}^{(i)^T}u)^2 = u^T\frac{1}{\mathrm{m}}\sum_{\mathrm{i}=1}^{\mathrm{m}}(\mathrm{x}^{(i)^T}x^{(i)})^2u = \lambda$$

$$\frac{1}{\mathrm{m}}\sum_{\mathrm{i}=1}^{\mathrm{m}}(\mathrm{x}^{(i)^T}x^{(i)})^2u = \lambda u$$



PCA

Object function of PCA is:

$$\mathrm{Max}\frac{1}{\mathrm{m}}\sum_{\mathrm{i}=1}^{\mathrm{m}}(\mathrm{x}^{(i)^T}u)^2$$

Dimension Reduction is similar with Classification, Cluster.

1. Model space
2. Object function
3. Optimization method

Multidimensional Scaling

From matrix decomposition's view :

$$XX^T = \mathrm{M} = U\Sigma U^{\mathrm{T}} \approx \sum_{i=1}^{d} \lambda_{\mathrm{i}} u_i \, u_i^{\mathrm{T}}$$

It is just a **Approximation** of M, so we can use something to replace M!!!! But what ?

Multidimensional Scaling

How is it like $X^T X$?

$$X^T X = \mathrm{M} = V \Sigma V^T \approx \sum_{i=1}^{d} \lambda_{\mathrm{i}} v_i \, v_i^{\mathrm{T}}$$

v_i^j is j-th data's value of i-th dimension in low-dimension representations.

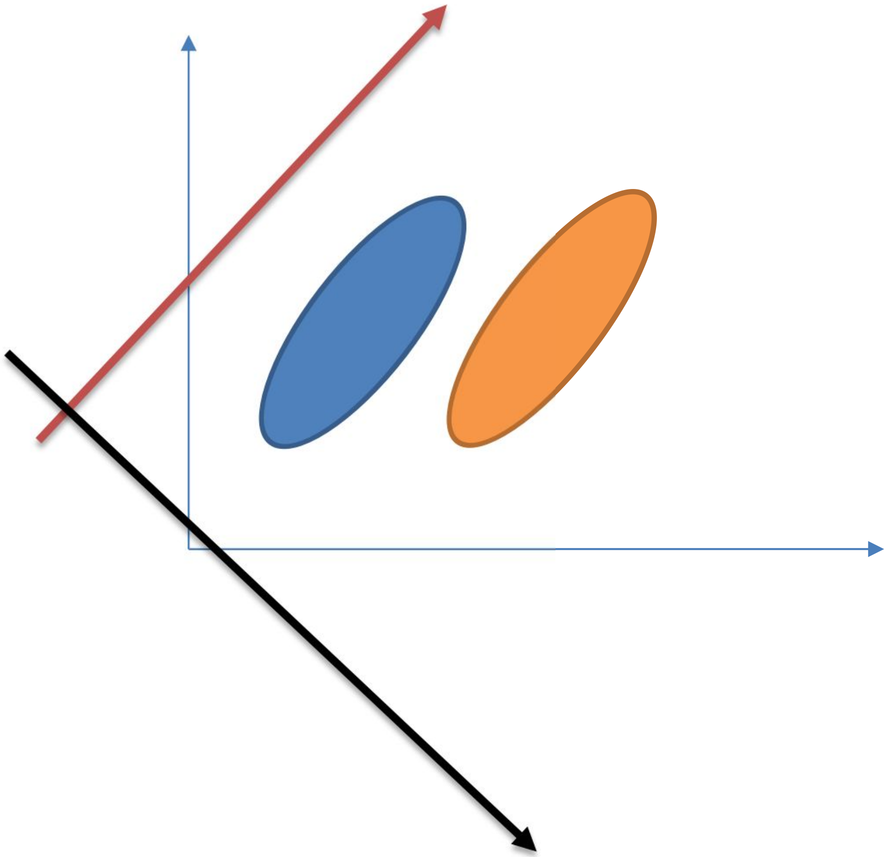
SVD

$$X = U\sqrt{\Sigma}V^{T}$$

$$XX^{T} = U\Sigma U^{\mathrm{T}}$$

$$X^{T}X = V\Sigma V^{T}$$

Linear Discriminant Analysis(LDA)



Is max variance best all the time???

Linear Discriminant Analysis(LDA)

数据挖掘实验室 Data Mining Lab

Target is to save information what can distinguish label of data.

$$J(w) = \frac{|\widetilde{m_1} - \widetilde{m_2}|^2}{\widetilde{s_1}^2 + \widetilde{s_2}^2}$$

$\widetilde{m_1}$: mean of class 1
$\widetilde{s_1}$: data variance of class 1

The intuition is to maximizes the projected class means while minimizing the classes variance in this direction

Factor analysis (FA)

Factor analysis is to use a potentially lower number of unobserved variables to describe observed variables.

The observed variables are modelled as linear combinations of the potential factors, plus "error" terms.

$$x_i - \mu_i = l_{i1}F_1 + \cdots + l_{ik}F_k + \varepsilon_i$$

In matrix terms, we have

$$x - \mu = LF + \varepsilon.$$

FA

Assumption on F

1. F and ε are independent.
2. $\mathrm{E}(F) = 0$.
3. $\mathrm{Cov}(F) = I$ (to make sure that the factors are uncorrelated).

Then we can compute covariance matrix of x

$$\Sigma = L\mathrm{Cov}(F)L^T + \mathrm{Cov}(\varepsilon)$$

$$\Sigma = LL^T + \Psi$$

FA

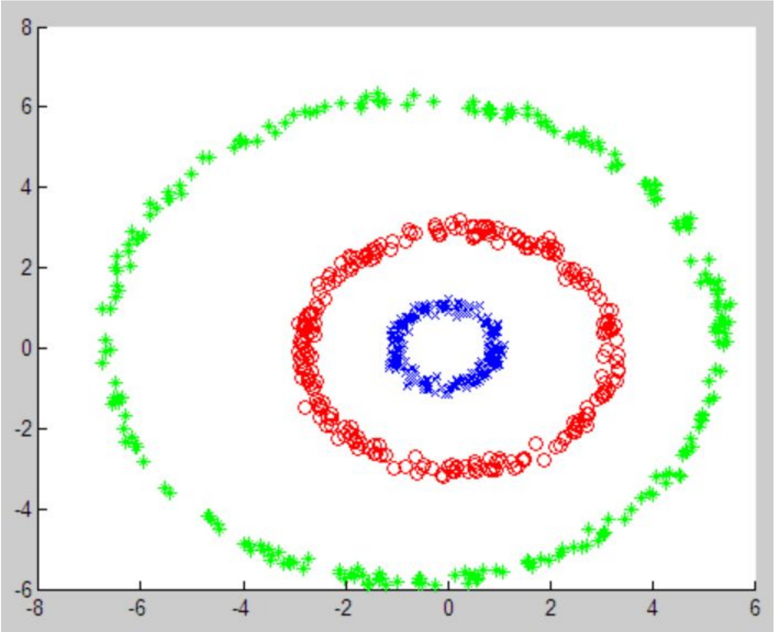
for any orthogonal matrix Q, if we set $L = LQ$, $F = Q^T F$

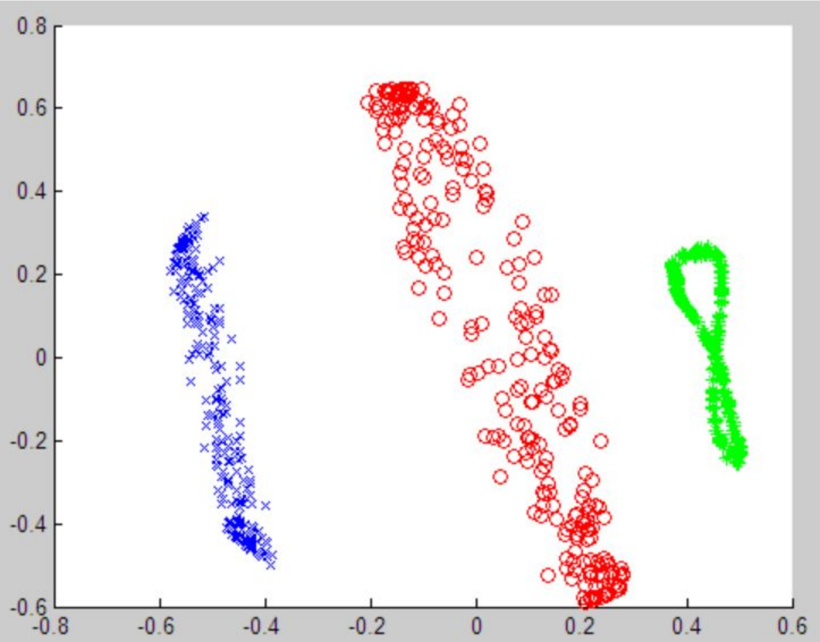
$$\Sigma = LQQ^T cov(F) QQ^T L^T + \Psi = LL^T + \Psi$$

This is factor rotation!!!

Kernel PCA

Kernel trick is to transform all calculation about dimension to calculation of inner product.





Kernel PCA

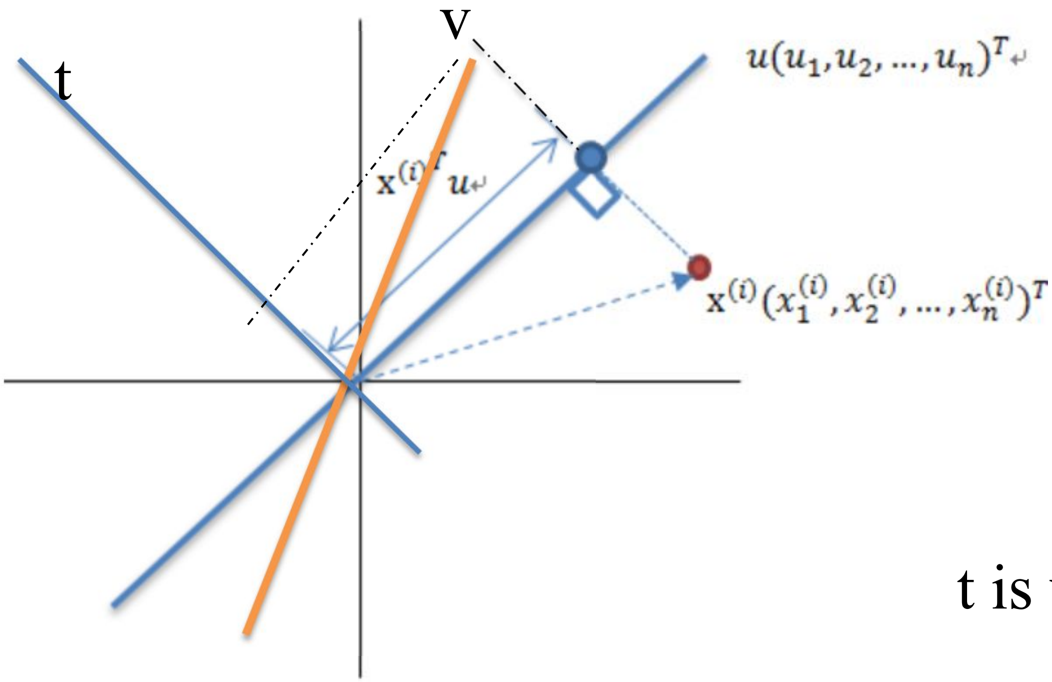
数据挖掘实验室 Data Mining Lab

$$\lambda_a \mathbf{u}_a = C\mathbf{u}_a = \frac{1}{N}\sum_i \mathbf{x}_i \mathbf{x}_i^T \mathbf{u}_a = \frac{1}{N}\sum_i (\mathbf{x}_i^T \mathbf{u}_a)\mathbf{x}_i$$

$$\mathbf{u}_a = \sum_i \frac{(\mathbf{x}_i^T \mathbf{u}_a)}{N\lambda_a}\mathbf{x}_i = \sum_i \alpha_i^a \mathbf{x}_i$$

every eigen-vector can be exactly written as some linear combination of the data-vectors

Kernel PCA



t is wasted.

Kernel PCA

$$\frac{1}{N}\Phi(X)\,\Phi(X)^T u = \lambda u$$

$$\frac{1}{N}\Phi(X)^T\Phi(X)\,\Phi(X)^T u = \lambda\Phi(X)^T u$$

$$\frac{1}{N}\Phi(X)^T\Phi(X)\,\Phi(X)^T\Phi(X)\,\alpha = \lambda\Phi(X)^T\Phi(X)\alpha$$

K is kernel Matrix of data , So

$$K^2\alpha = N\lambda K\alpha$$

$$K\alpha = N\lambda\alpha$$

We just need use pca on K.

Kernel PCA

how center data in feature space?

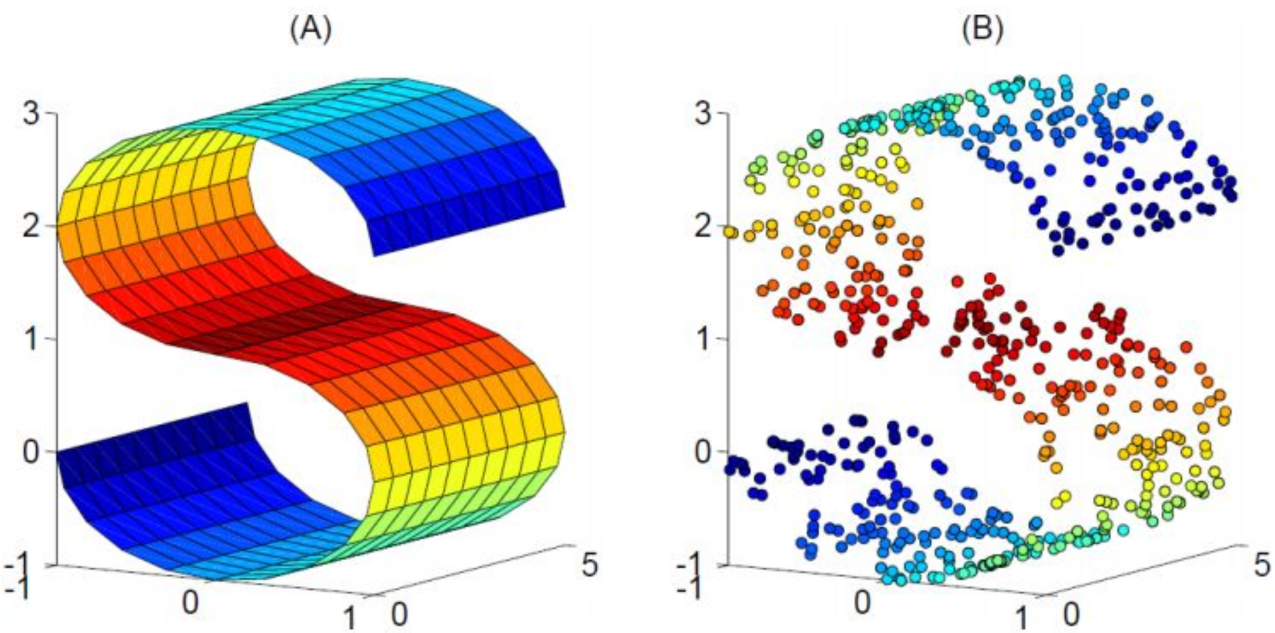
$$\Phi_i = \Phi_i - \frac{1}{N}\sum_k \Phi_k$$

$$\mathrm{K_{ij}} = (\Phi_i - \tfrac{1}{N}\textstyle\sum_k \Phi_k)(\Phi_j - \tfrac{1}{N}\textstyle\sum_k \Phi_k)^T$$

So K can be computed by inner product of Φ_i(i=1…k)

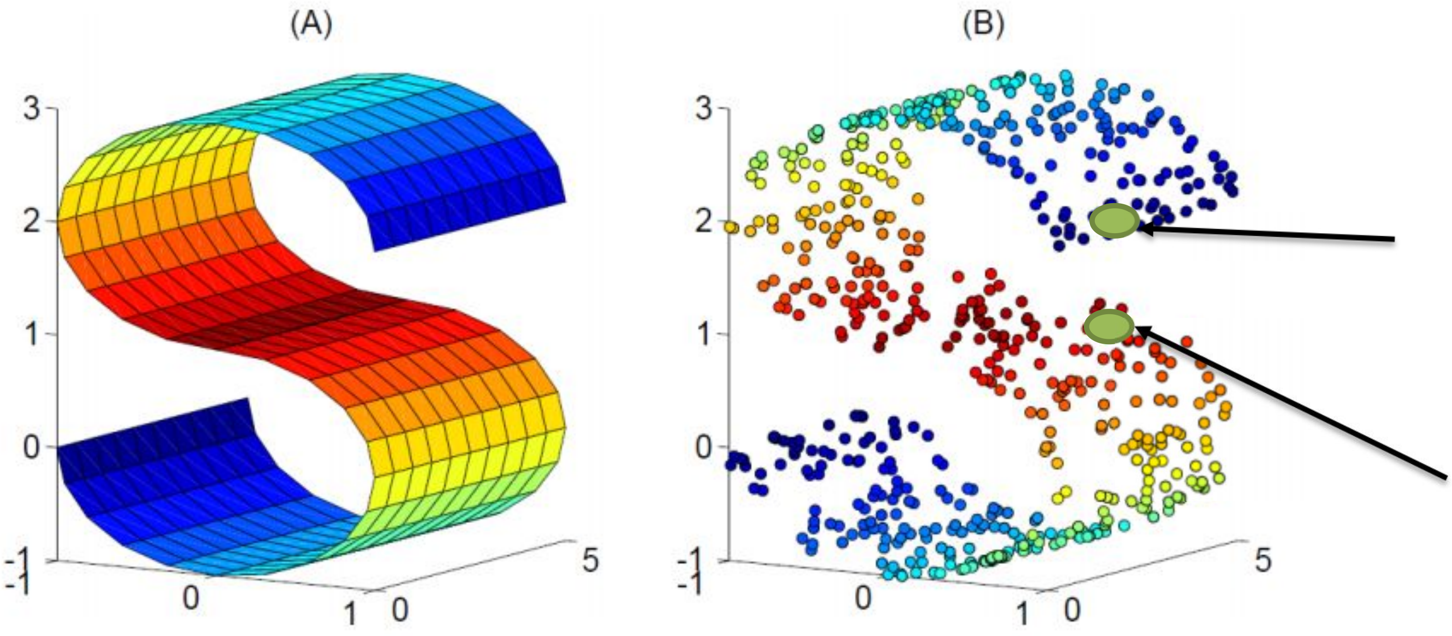
What is **Manifold**?

a low dimensional hyperplane embedded in a higher dimensional space



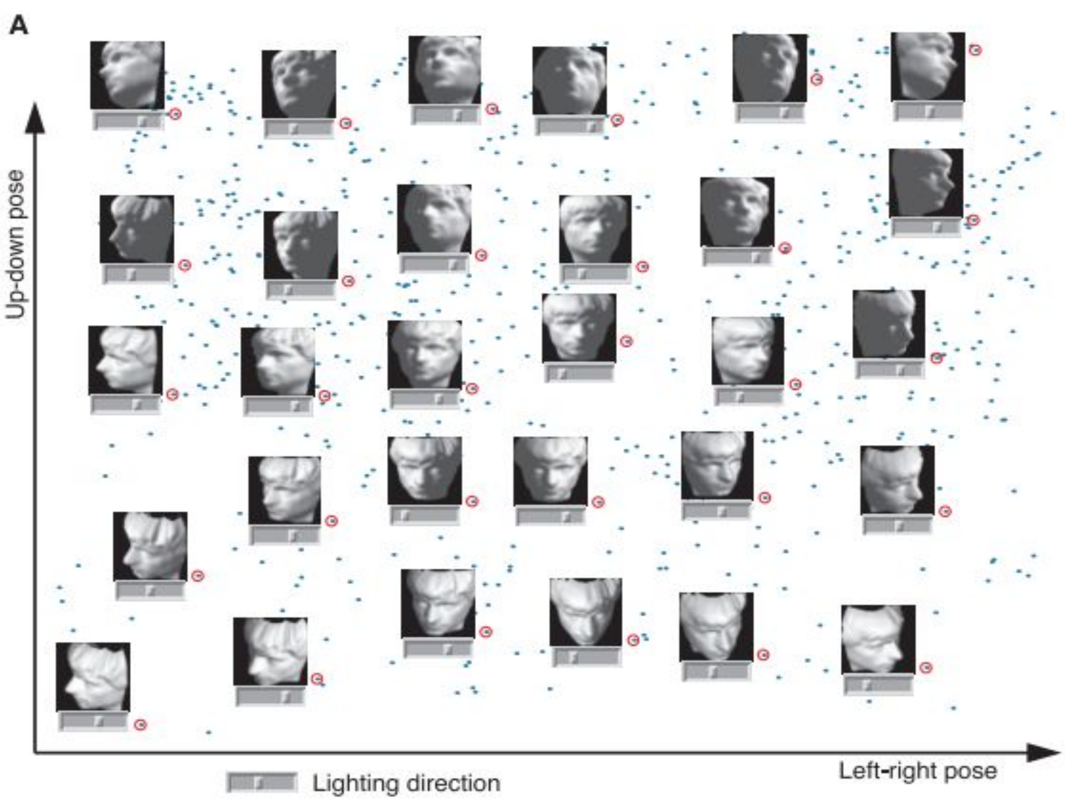
What is Manifold learning's target

In my opinion, manifold learning is to save local proximity relationships using low-dimensional representations.



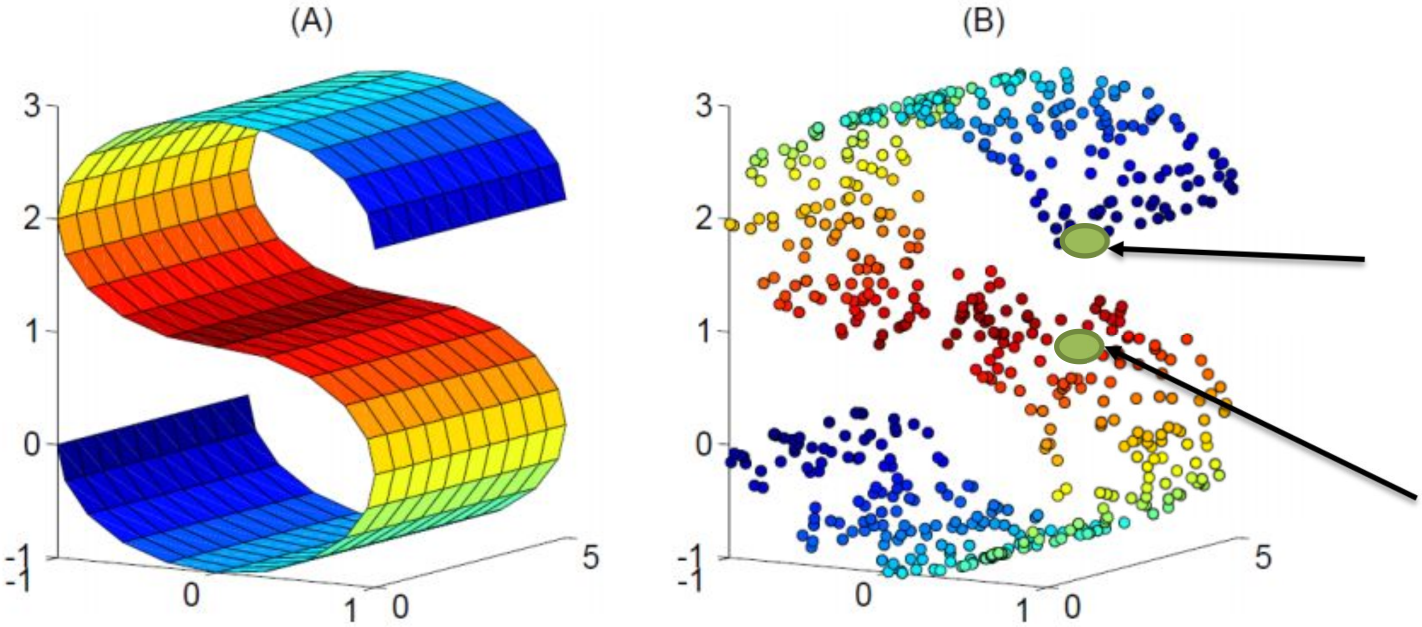
ISOMAP

Our goal is to discover, given only the unordered high-dimensional inputs, low-dimensional representations that capture the intrinsic degrees of freedom of a data set.



ISOMAP

Global distance can be approximated by
adding up a sequence of “short hops” between neighboring points.



ISOMAP

1. for each data point x_i compute distance with its k neighborhood.
2. Compute shortest paths according step 1's result.
3. Use MDS

Locally Linear Embedding

Its assumption is data point and its neighbors to lie on or close to a locally linear patch of the manifold.

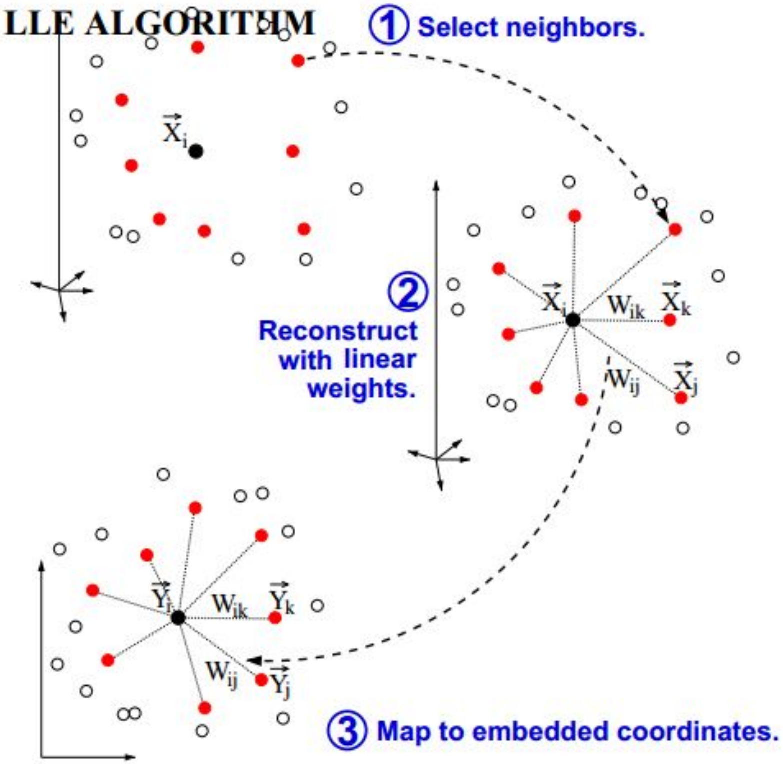
$$x_i = \sum_{x_j \in N_i} W_{i,j} x_j \qquad \text{s.t} \ \sum_{j=1}^{n} W_{i,j} = 1 \qquad (1)$$

LLE's target is to save local linear relation information in low-dimension representations.

Locally Linear Embedding

LLE ALGORITHM

1. **Compute the neighbors of each data point, $\vec{X}_i$.**
2. **Compute the weights W_{ij} that best reconstruct each data point $\vec{X}_i$ from its neighbors, minimizing the cost in Equation (1) by constrained linear fits.**
3. **Compute the vectors $\vec{Y}_i$ best reconstructed by the weights W_{ij}, minimizing the quadratic form in Equation (2) by its bottom nonzero eigenvectors.**



$$\Phi(Y) = \sum_i \left| \vec{Y}_i - \Sigma_j W_{ij} \vec{Y}_j \right|^2 \qquad (2)$$

LLE

$$(\mathrm{y} - \mathrm{Wy})^T(\mathrm{y} - \mathrm{Wy}) = \lambda$$

$$y^{\mathrm{T}}(\mathrm{I} - \mathrm{W})^T(\mathrm{I} - \mathrm{W})\mathrm{y} = \lambda$$

$$(\mathrm{I} - \mathrm{W})^T(\mathrm{I} - \mathrm{W})\mathrm{y} = \lambda \mathrm{y}$$

compute $(\mathrm{I} - \mathrm{W})^T(\mathrm{I} - \mathrm{W})$ minimum eigenvalue!!!!

Spectral cluster

数据挖掘实验室 Data Mining Lab

1.Construct the similarity matrix W

$$
\begin{bmatrix} 0 & W_{12} & W_{13} \\ W_{12} & 0 & W_{23} \\ W_{13} & W_{23} & 0 \end{bmatrix}
$$

2. Construct the Laplacian Matrix L

$$
L = \begin{bmatrix} W_{12}+W_{13} & 0 & 0 \\ 0 & W_{23}+W_{12} & 0 \\ 0 & 0 & W_{23}+W_{13} \end{bmatrix} - \begin{bmatrix} 0 & W_{12} & W_{13} \\ W_{12} & 0 & W_{23} \\ W_{13} & W_{23} & 0 \end{bmatrix}
$$

Spectral cluster

3. Compute the first k eigenvectors $u_1, \ldots, u_k$ of L

4. For i = 1, . . . , n, let y_i be the vector corresponding to the i-th row of U.

5. Cluster the points $y_i (i=1,\ldots,n)$ with the k-means algorithm into clusters

Laplacian Eigenmaps

Just save local proximity information

$$\min \sum_{i,j} (y_i - y_j)^2 W_{ij}$$

The intuition is "if x_i and x_j are more similar, then y_i and y_j are more similar"

$$\min \quad \frac{1}{2} \sum_{i,j} (y_i - y_j)^2 W_{ij} = \mathbf{y}^T L \mathbf{y}$$

Laplacian Eigenmaps

$$\min \quad \frac{1}{2}\sum_{i,j}(y_i - y_j)^2 W_{ij} = \mathbf{y}^T L \mathbf{y}$$

$$\mathrm{Ly} = \mathrm{y}\lambda$$

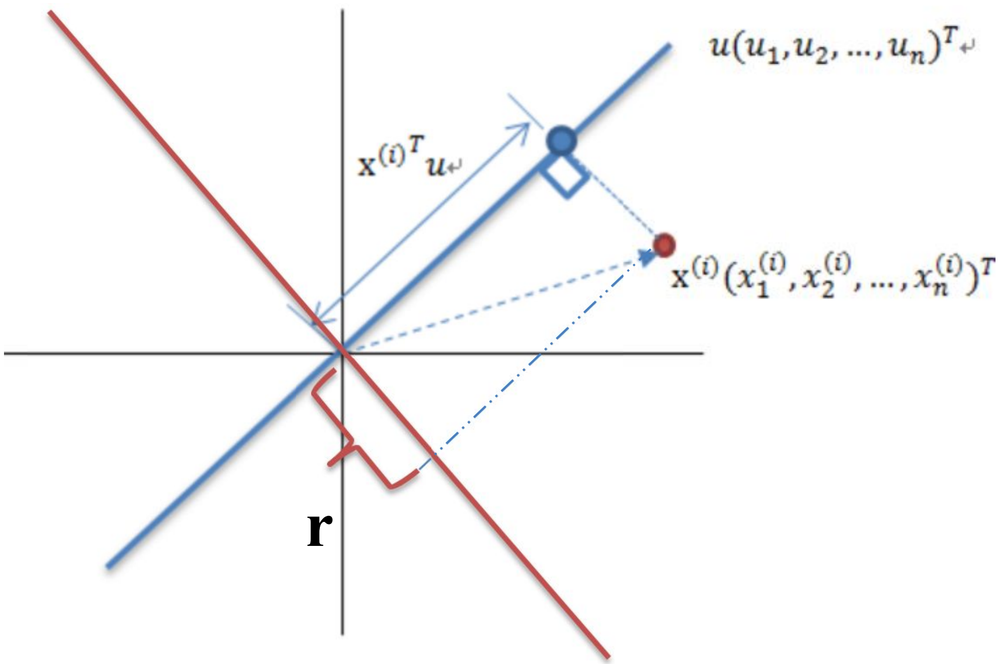
This is also a problem of computing eigenvalue of L.

LE & Spectral cluster

Spectral cluster just use LE to reduce dimension and then use k-means to cluster.

My small idea

Combine residual vector and data of subspace



Thanks