Knowledge Discovery & Data Mining

— Data Preprocessing —

Dimensionality Reduction: Feature Extraction

Instructor: Yong Zhuang

yong.zhuang@gvsu.edu

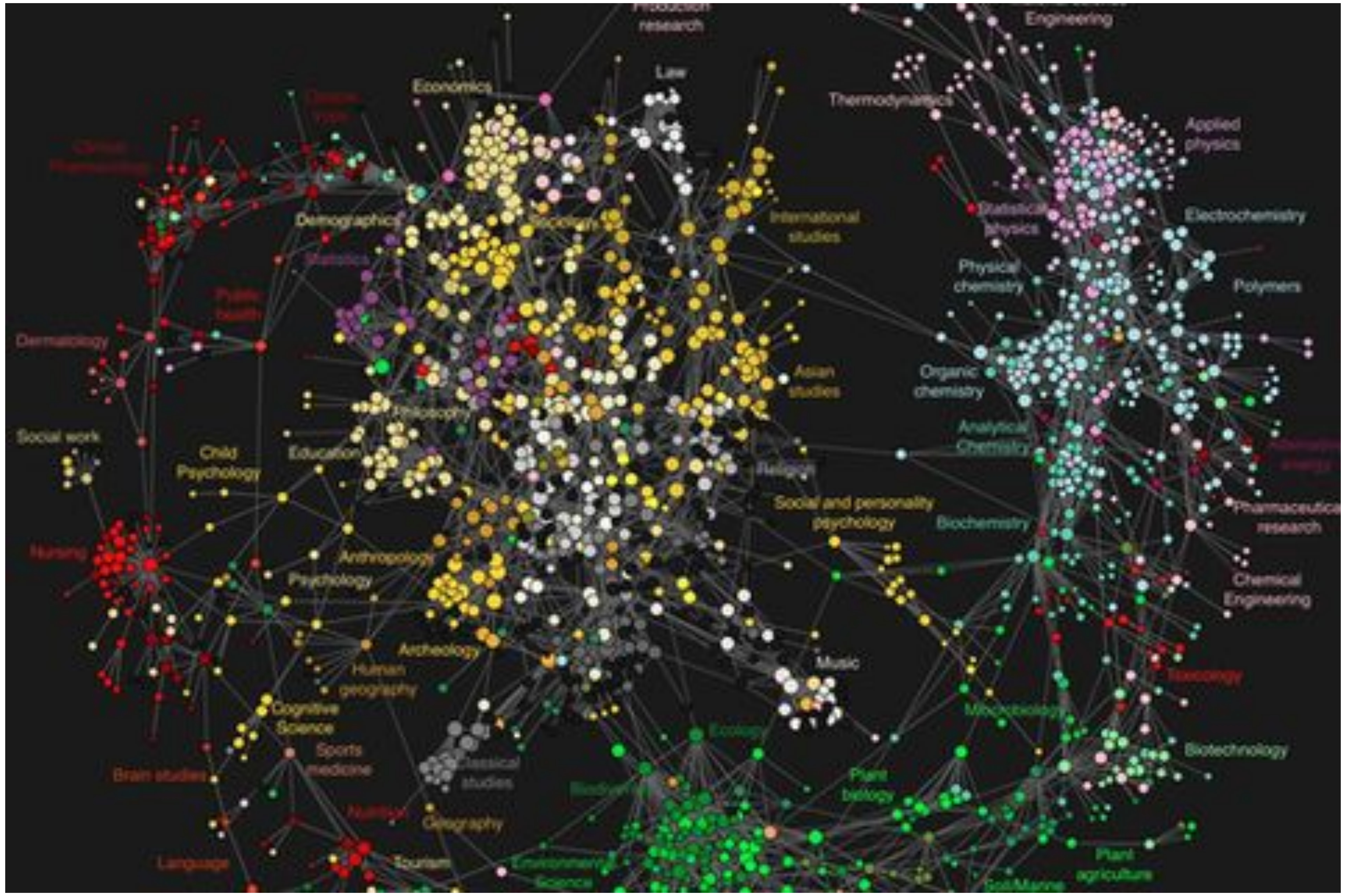
Outline

- Dimension Reduction
    - Curse of Dimensionality
    - Feature extraction
        - Principal components analysis(PCA)
        - Kernel PCA
        - Stochastic neighbor embedding(SNE)

Curse of Dimensionality

Dimensionality: refers to the number of features or attributes within a dataset.

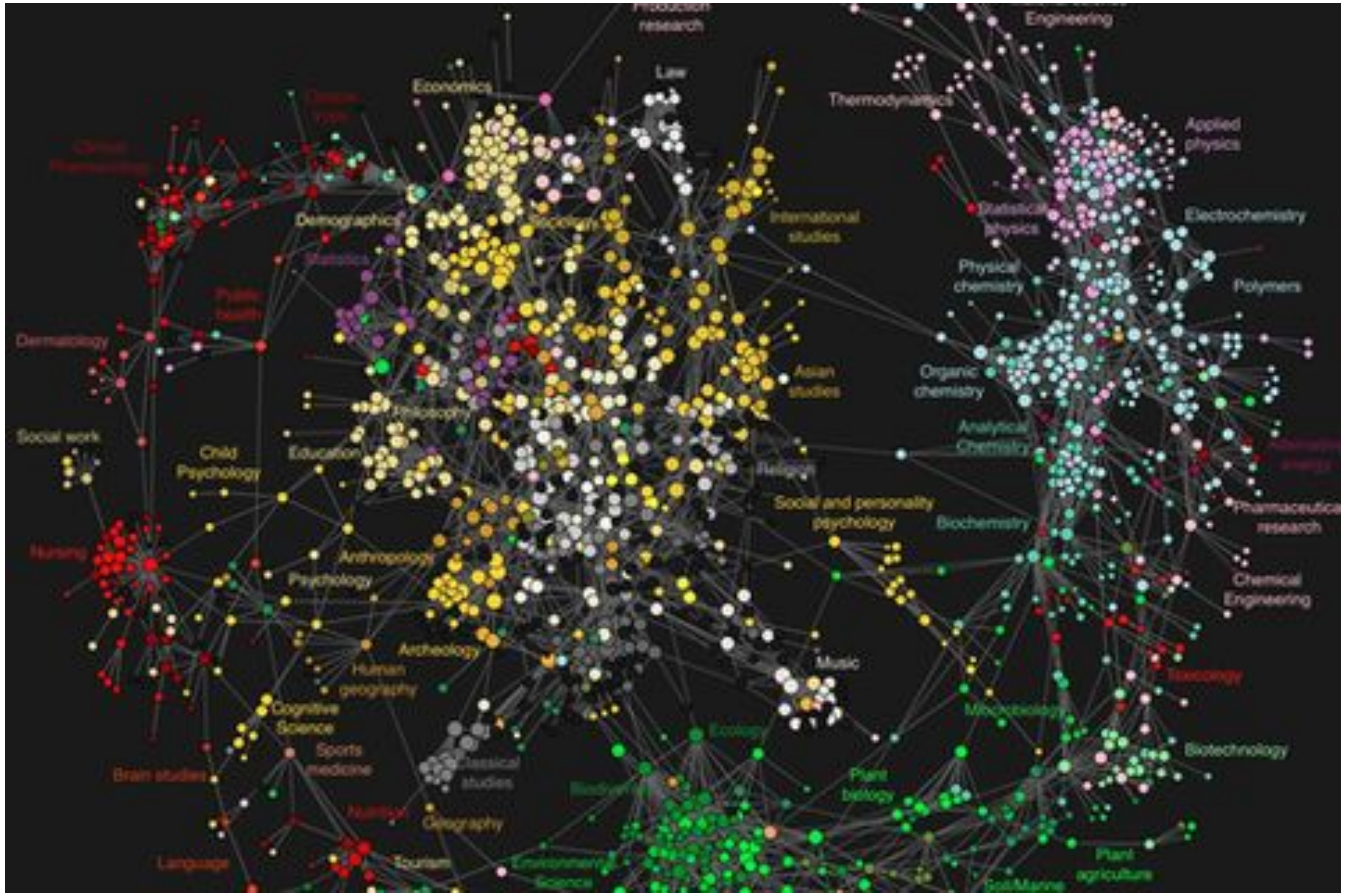
*When the number of features significantly exceeds the number of observations, many algorithms can struggle to effectively train models. This is called the "**Curse of Dimensionality**," and it especially impacts data mining algorithms that depend on distance calculations, as it can hinder the effective training of models.*



Curse of Dimensionality

How to Solve the Curse of Dimensionality?

Dimension Reduction



Dimension Reduction

Dimension Reduction: It's a process that reduces the number of random variables under consideration by obtaining a set of principal variables that retain the most important information in the data while discarding the redundant or less important features.

Feature extraction: Transforms data into a set of new features.

- **Method:** PCA, Kernel PCA, Stochastic neighbor embedding, Autoencoders, ….
- **Advantages:** The newly derived features can capture essential information in fewer dimensions.

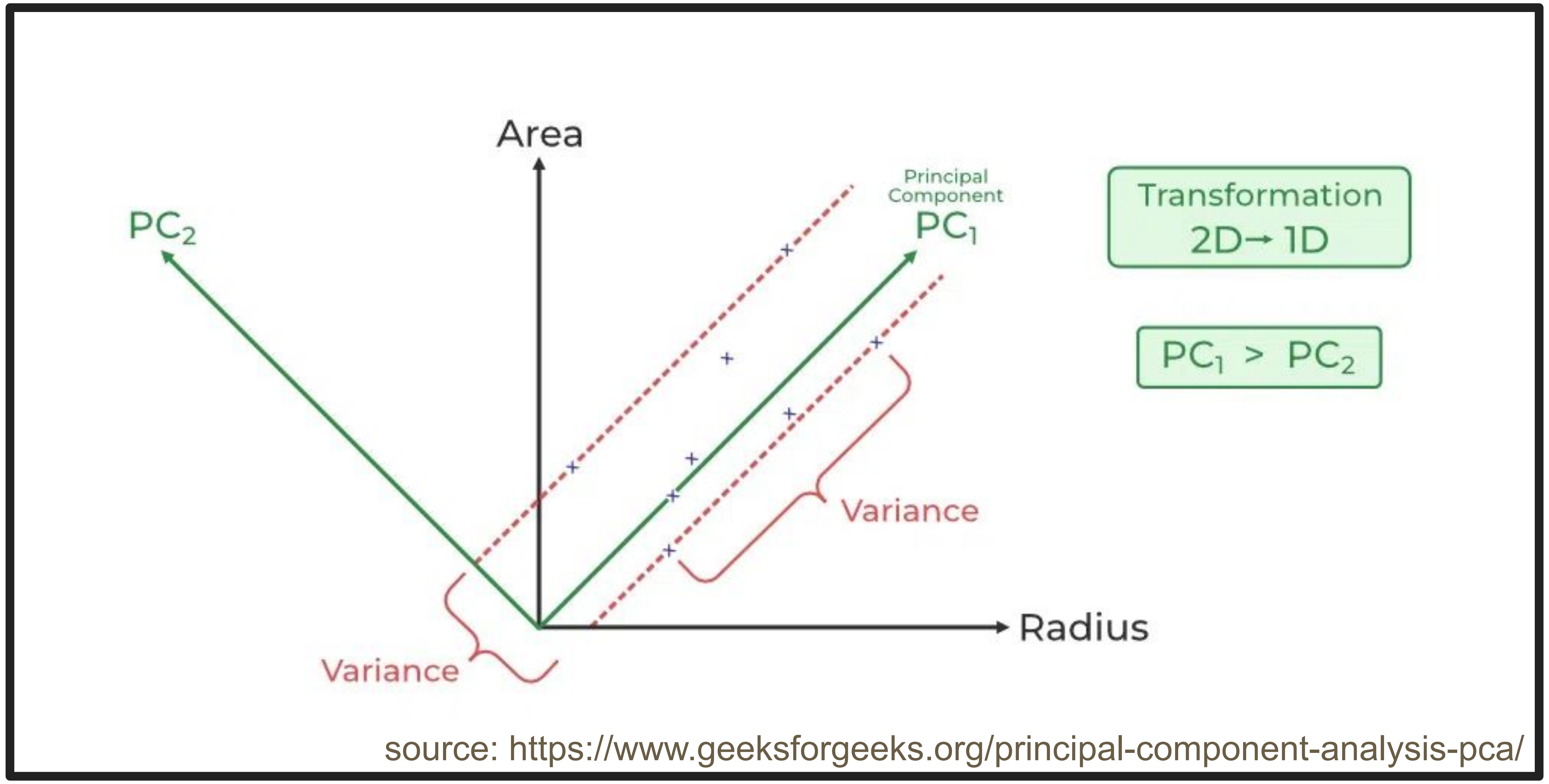
Feature selection: Selects a subset of the most relevant features for model construction.

- **Method:** Filter methods, wrapper methods, embedded methods.
- **Advantages:** Enhances model interpretability, discards irrelevant or redundant features.

Feature Extraction: PCA

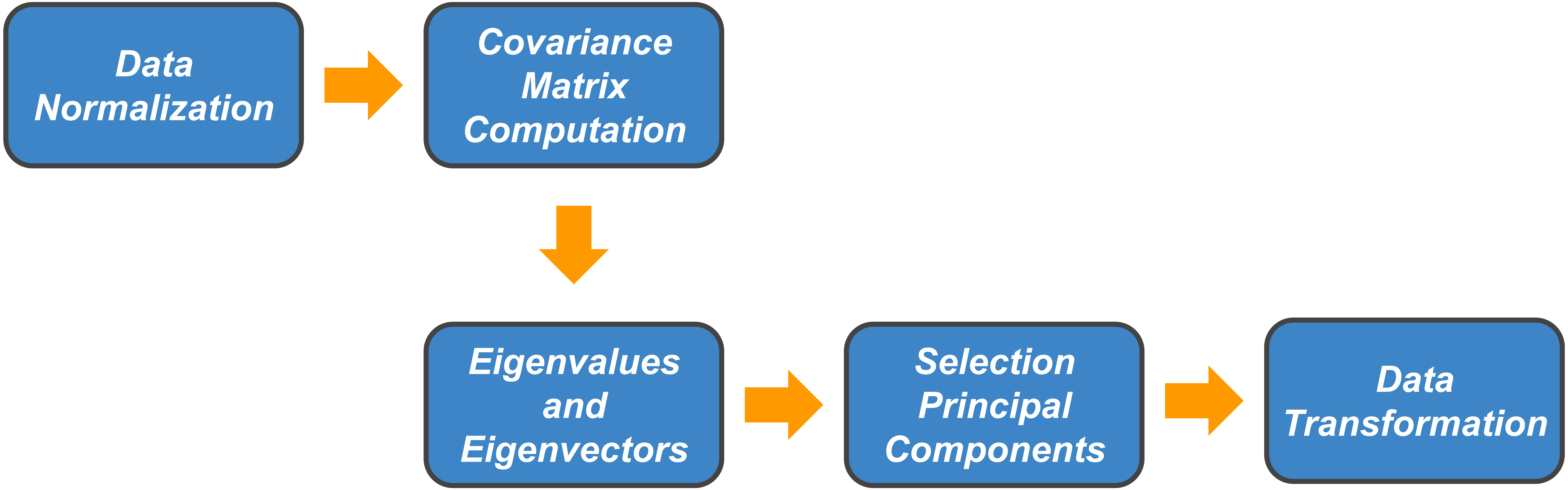
Principal Component Analysis (PCA) is a statistical procedure that uses an orthogonal transformation that converts a set of correlated variables to a set of uncorrelated variables. it is the most widely used tool in exploratory data analysis and in data mining for predictive models.

PCA identifies a set of orthogonal axes, called ***principal components***, that capture the maximum ***variance*** in the data. The principal components are linear combinations of the original variables in the dataset and are ordered in decreasing order of importance. The total variance captured by all the principal components is equal to the total variance in the original dataset.



source: https://www.geeksforgeeks.org/principal-component-analysis-pca/

Feature Extraction: PCA



Data Normalization

Covariance Matrix Computation

Eigenvalues and Eigenvectors

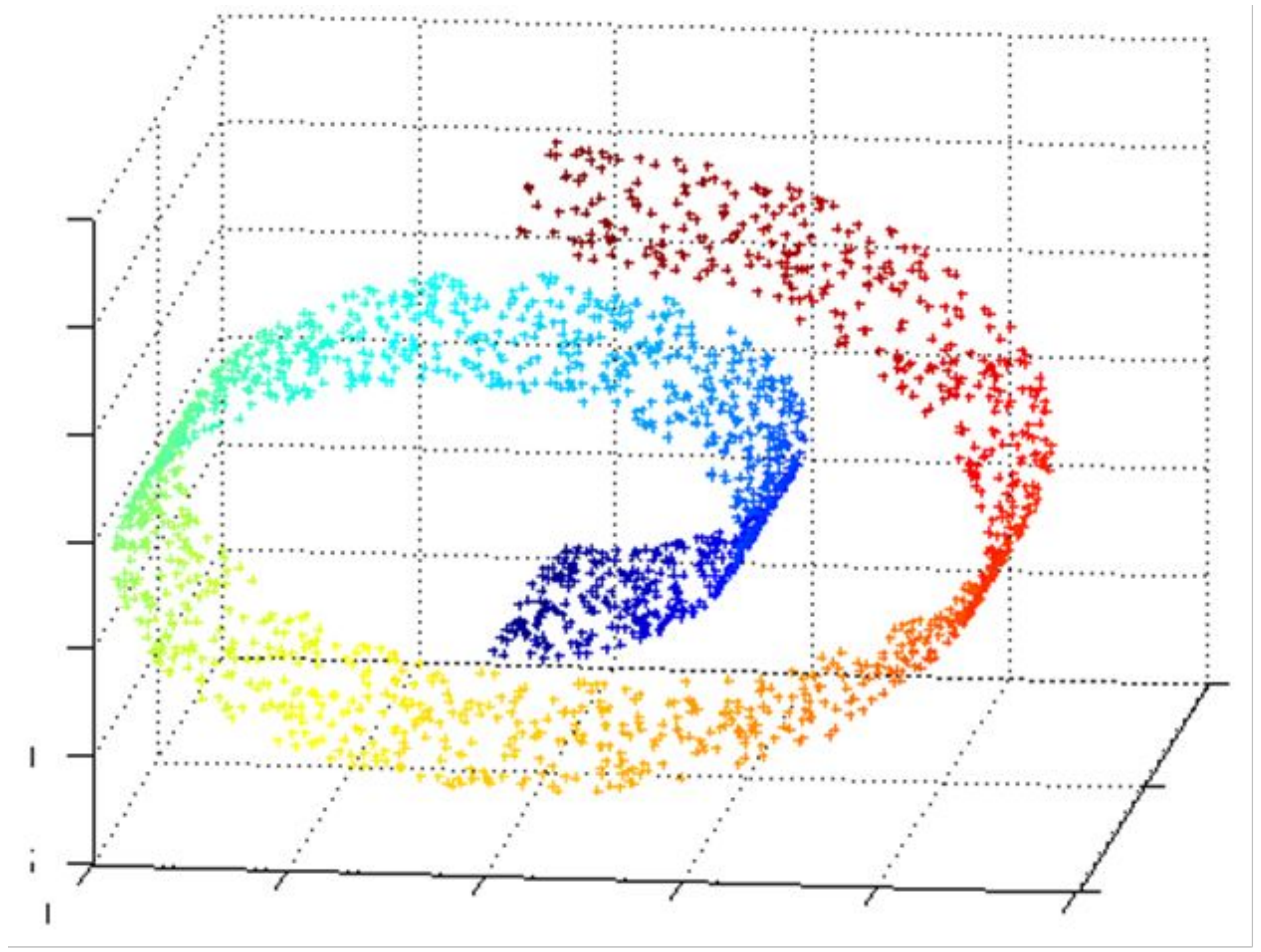
Selection Principal Components

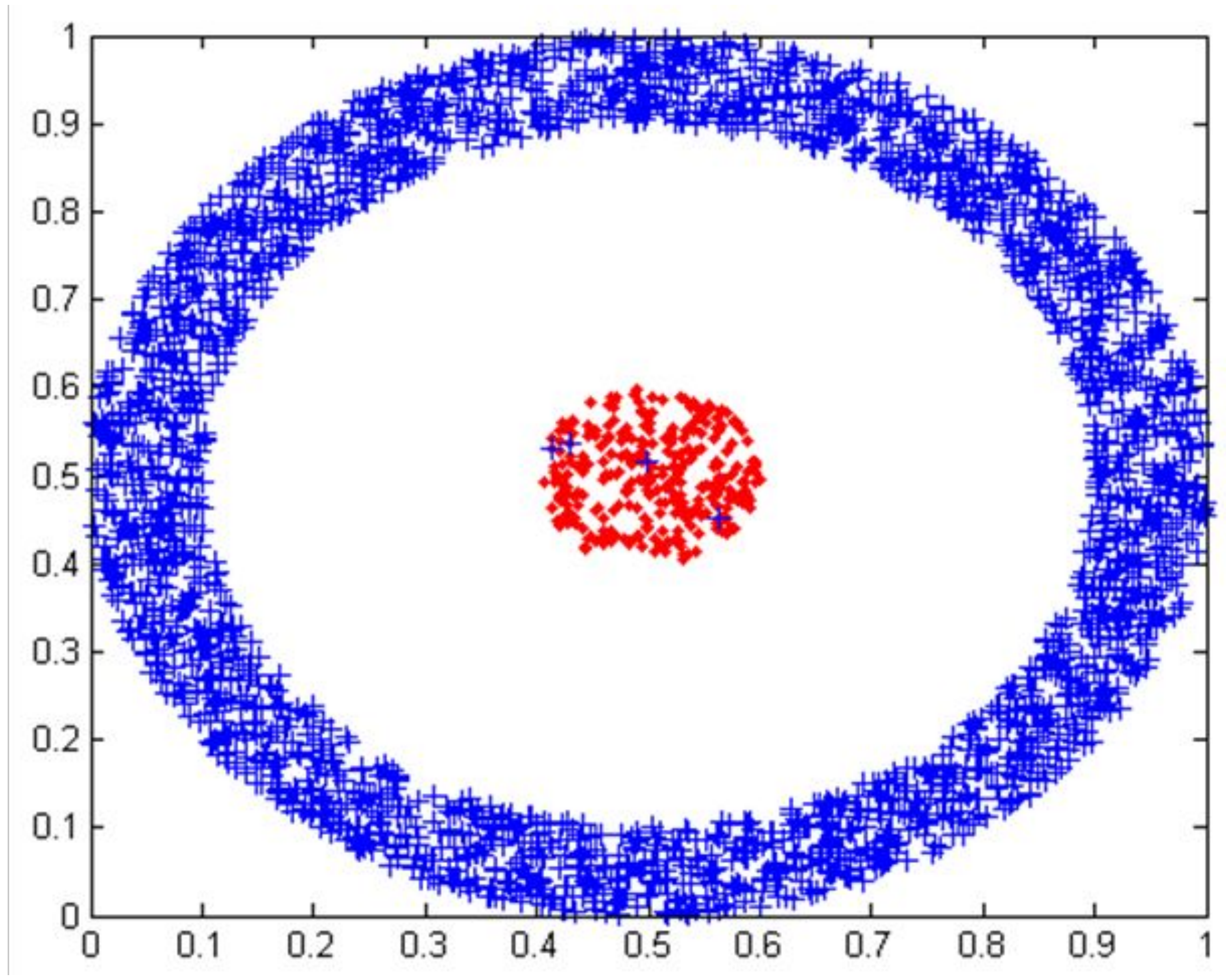
Data Transformation

Nonlinear Feature Extraction Methods

PCA is a linear method for dimensionality reduction in that each principal component is a linear combination of the original input attributes. This works well if the input data approximately follows a Gaussian distribution or forms a few linearly separable clusters. When the input data are linearly inseparable, PCA becomes ineffective.

Nonlinear Feature Extraction Methods





Nonlinear Feature Extraction: General procedure

Suppose there are n data tuples **x**$_i$, (i = 1, ..., n), each of which is represented by a ***d***-dimensional attribute vector.

How can we reduce the dimensionality to k where k << d?

Two steps.

1. **Constructing proximity matrix**: we construct an n×n proximity matrix P whose entry P(i,j) (i,j = 1, ..., n) indicates the affinity or relevance between the two corresponding data tuples xi and xj .
2. **Preserving proximity**: we learn the new, low-dimensional representations of the input data tuples in the k-dimensional space $\hat{x}_i$ (i = 1, ..., n) so that the proximity matrix P constructed in the first step is somewhat preserved.

Kernel PCA

1. we use a kernel function κ(·) to construct the proximity matrix, called kernel matrix.
   a. a kernel function computes the similarity of a pair of input data tuples in some high-dimensional, often nonlinear, space.
2. we estimate proximity (i.e., similarity) in low-dimensional space based on the learned low dimensional representations: $\hat{P}(i, j) = \hat{\boldsymbol{x}}_i \cdot \hat{\boldsymbol{x}}_j, (i, j = 1, \ldots, n)$ where · is the vector inner product.

$\hat{P}$ is as close as possible to the kernel matrix P

↓

minimize $\sum_{i,j=1}^{n} (P(i, j) - \hat{P}(i, j))^2 = \underline{\| P - \hat{P} \|_{fro}^2}$

Frobenius norm

Kernel PCA

Typical choices for the kernel functions

- polynomial kernel: $\kappa(\boldsymbol{x}_i, \boldsymbol{x}_j) = (1 + \boldsymbol{x}_i \cdot \boldsymbol{x}_j)^p$
- radial basis function (RBF):

$$\kappa(\boldsymbol{x}_i, \boldsymbol{x}_j) = e^{\frac{-\|\boldsymbol{x}_i - \boldsymbol{x}_j\|^2}{2\sigma^2}}$$

- linear kernel: $\kappa(\boldsymbol{x}_i, \boldsymbol{x}_j) = \boldsymbol{x}_i \cdot \boldsymbol{x}_j$ → KPCA = PCA

Stochastic neighbor embedding(SNE)

1. we first construct the proximity matrix P as follows:

$$P(i,j) = \frac{e^{-d_{ij}^2}}{\sum_{l=1, l\neq i}^{n} e^{-d_{il}^2}}, \text{ where } d_{ij}^2 = \frac{\|\boldsymbol{x}_i - \boldsymbol{x}_j\|^2}{2\sigma^2} \text{ and } \sigma \text{ is the parameter}$$

   a. P(i,j): the probability that data tuple xj is the neighbor of data tuple xi
   b. the closer the two data tuples are (i.e., smaller dij), the more likely xj is the neighbor of xi

2. We estimate proximity matrix in low-dimensional space in the similar way:

$$\hat{P}(i,j) = \frac{e^{-\|\hat{\boldsymbol{x}}_i - \hat{\boldsymbol{x}}_j\|^2}}{\sum_{l=1, l\neq i}^{n} e^{-\|\hat{\boldsymbol{x}}_i - \hat{\boldsymbol{x}}_l\|^2}}$$

$\hat{P}$ be as close as possible to the proximity matrix P: $P \approx \hat{P}$

Stochastic neighbor embedding(SNE)

each row of matrices P and $\hat{P}$ is a probability distribution that tells the probability that each data tuple is the neighbor of a give data tuple.

$\hat{P}$ be as close as possible to the proximity matrix P: $\boxed{P \approx \hat{P}}$

Stochastic neighbor embedding(SNE)

each row of matrices P and $\hat{P}$ is a probability distribution that tells the probability that each data tuple is the neighbor of a give data tuple.

$\hat{P}$ be as close as possible to the proximity matrix P: $\boxed{P \approx \hat{P}}$

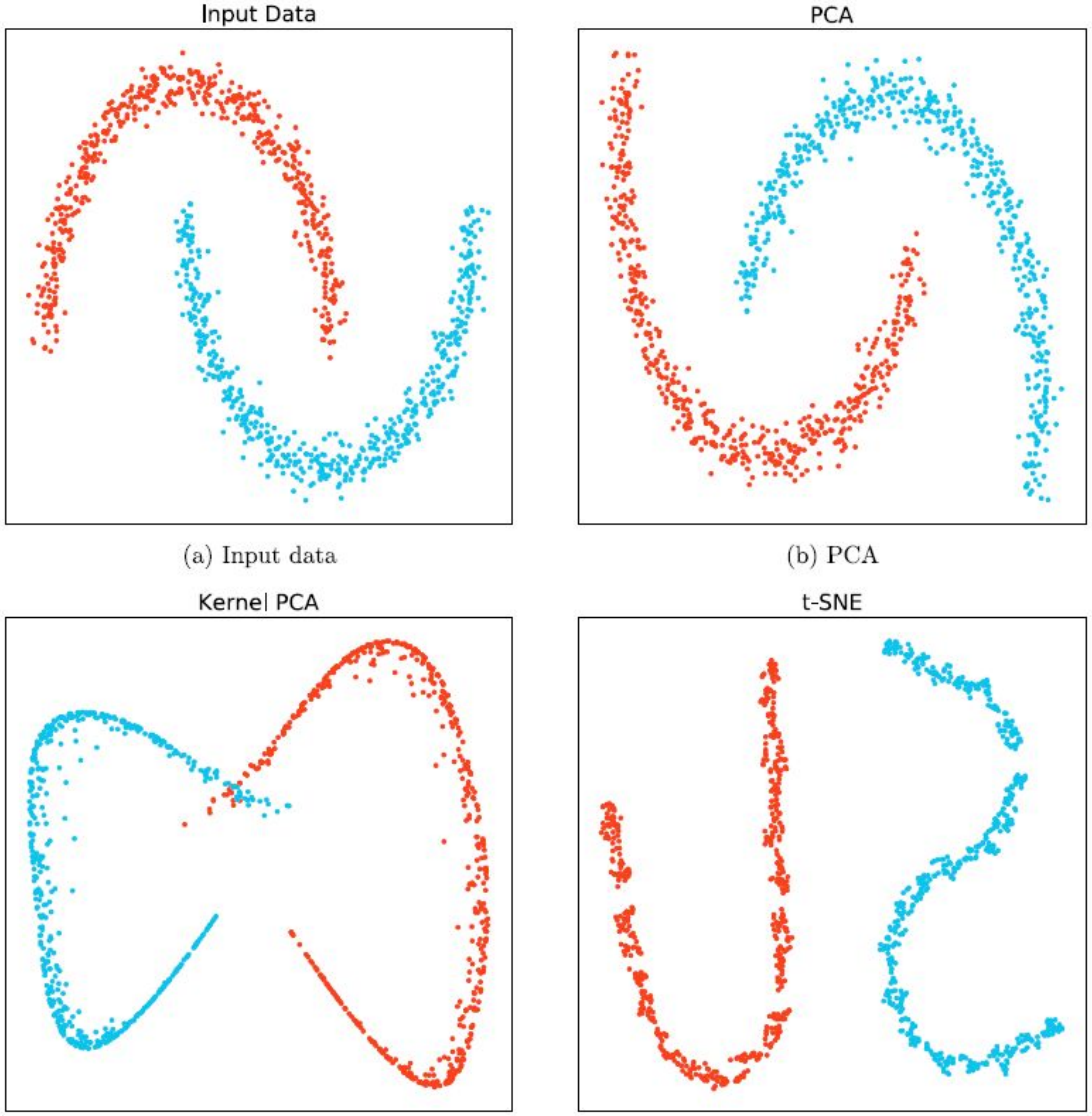
KL divergences

$$\hat{\boldsymbol{x}}_i = \arg\min_{\hat{\boldsymbol{x}}_i,(i=1,\ldots n)} \sum_{i=1}^{n} D_{KL}(P_i||\hat{P}_i)$$, where P_i and $\hat{P}_i$ are the ith rows of P and $\hat{P}$

A variant of SNE named t-SNE (t-distributed stochastic neighbor embedding) has been widely used to project the multi-dimensional representation produced by various deep learning models. **Artworks tSNE map**

Nonlinear dimensionality reduction methods

Example. Given a collection of data tuples in 2-D space. The input data naturally form two clusters: one crescent shape facing up and one facing down. These two clusters are entangled with each other, and there is no way we can find a linear subspace (a linear line in this case) to separate them from each other. This means that no matter what kind of line we choose from the input space, if we project the original data tuples onto this line, the projected portions (i.e., the low-dimensional representation) will always be mixed with each other.



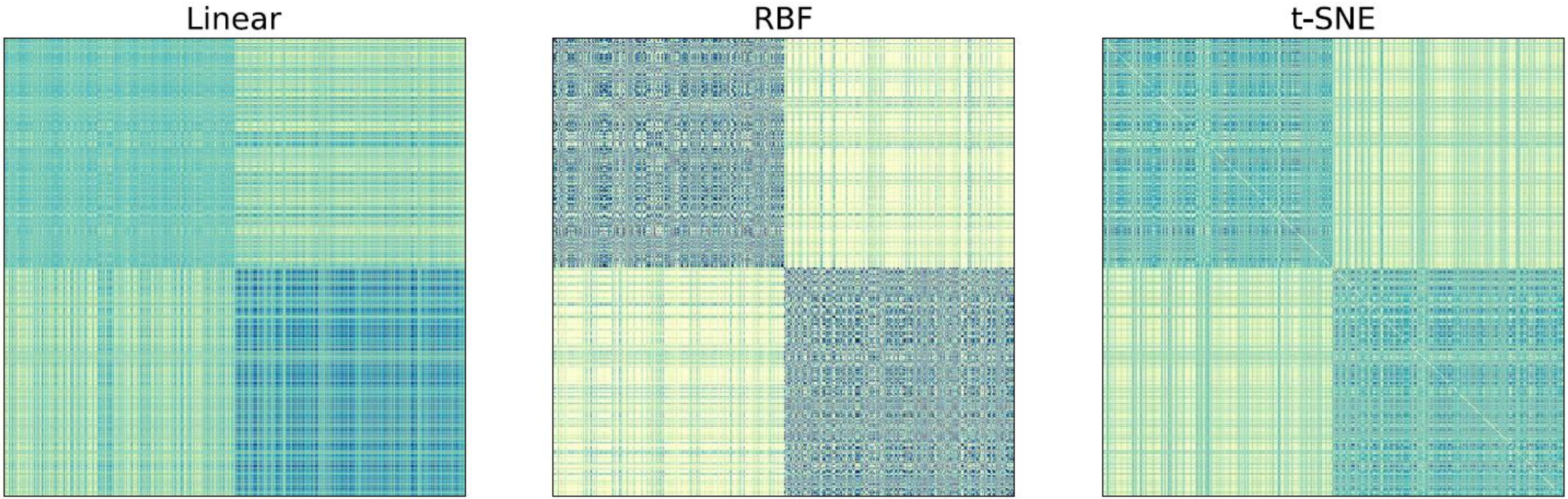
(a) Input data

(b) PCA

(c) KPCA

(d) t-SNE

Nonlinear dimensionality reduction methods



(a) PCA

(b) KPCA

t-SNE

Summary

- Dimension Reduction
  - Curse of Dimensionality
  - Feature extraction
    - Principal components analysis(PCA)
    - Kernel PCA
    - Stochastic neighbor embedding(SNE)
- Sample Code