Manifold Learning Study Notes

Shao Yuanlong

Definition

$\mathbf{x}_i$, $1 \le i \le n$, are vectors at D dimensional spaces

$\mathbf{y}_i$, $1 \le i \le n$, are vectors at d dimensional spaces, and d << D

The problem is to find a functional projection

$$\mathbf{y}_i = f(\mathbf{x}_i)$$

So that the yielded low dimensional space has properties required for the application.

Linear Projection Approach

$\mathbf{y}_i = \mathbf{W}^T \cdot \mathbf{x}_i$, here both $\mathbf{x}_i$ and $\mathbf{y}_i$ are with zero means. $\mathbf{W} = [\mathbf{e}_1 \cdots \mathbf{e}_d]$, $\|\mathbf{e}_i\| = 1$

Principle Component Analysis (PCA)

Objective

$$\max \sum_{i=1}^{n} \|\mathbf{y}_i\|^2 \quad \text{or} \quad \min \sum_{i=1}^{n} \|\mathbf{W} \cdot \mathbf{y}_i - \mathbf{x}_i\|^2$$

So PCA can be regarded as maximizing the scattering of the target space or finding the best hyper planes for the projected space to represent the original data, which is also called the principle components.

Solution

$$\text{For each } j,\ \text{with } 1 \le j \le d \begin{cases} \max \left\| \mathbf{e}_j{}^T \sum_{i=1}^{n} \mathbf{x}_i \mathbf{x}_i{}^T \mathbf{e}_j \right\|^2 \\ s.t.\ \ \mathbf{e}_j{}^T \mathbf{e}_j = 1 \end{cases}$$

$$\Rightarrow \left(\sum_{i=1}^{n} \mathbf{x}_i \mathbf{x}_i{}^T \right) \mathbf{e}_j = \lambda_j \mathbf{e}_j$$

So PCA can be solved by calculating eigenvectors of the scattering matrix.

Fisher's Linear Discriminant Analysis (LDA)

Objective

Find the low dimensional representative hyper planes that are most effective for discrimination.

Derivation

n samples totally within k classes, n_i samples for each class C_i

$$\mathbf{S}_B = \sum_{i=1}^{n} \overline{\mathbf{x}}_{c(i)} \overline{\mathbf{x}}_{c(i)}^T = \sum_{i=1}^{k} n_i \overline{\mathbf{x}}_i \overline{\mathbf{x}}_i^T$$

$$\mathbf{S}_W = \sum_{i=1}^{n} \left(\mathbf{x}_i - \overline{\mathbf{x}}_{c(i)} \right) \cdot \left(\mathbf{x}_i - \overline{\mathbf{x}}_{c(i)} \right)^T$$

$$\max \frac{\left\| \mathbf{W}^T \mathbf{S}_B \mathbf{W} \right\|}{\left\| \mathbf{W}^T \mathbf{S}_W \mathbf{W} \right\|}$$

$$\Rightarrow \mathbf{S}_B \mathbf{e}_j = \lambda_j \mathbf{S}_W \mathbf{e}_j$$

So LDA can be solved as a generalized eigen problem.

Why "c-1" classes at Sec 3.8.3 from the book "Pattern Classification"?

Local Discriminant Embedding (LDE)

Objective

Maximize the distances of vectors within neighborhood range.

Minimize the distances of vectors without neighborhood range.

Derivation

Some of the steps are inspired from Laplacian Eigenmaps and LPP

Step1. Build neighbor graph using kNN or ϵ-ball.

Step2. Calculate weight matrices

$$\mathbf{M} = \{m_{ij}\}$$
$$\mathbf{M}' = \{m_{ij}'\}$$
$$m_{ij} = \begin{cases} e^{-\|\mathbf{x}_i - \mathbf{x}_j\|/t}, & \text{if } \langle i, j\rangle \text{ is an edge} \\ 0, & \text{otherwise} \end{cases}$$
$$m_{ij}' = \begin{cases} e^{-\|\mathbf{x}_i - \mathbf{x}_j\|/t}, & \text{if } \langle i, j\rangle \text{ isn't an edge} \\ 0, & \text{otherwise} \end{cases}$$

Step3. Solve the optimization for the projection matrix $\mathbf{W}$:

$$\max \sum_{i,j} \left\|\mathbf{W}^T\mathbf{x}_i - \mathbf{W}^T\mathbf{x}_j\right\|^2 m_{ij}'$$
$$s.t. \quad \sum_{i,j} \left\|\mathbf{W}^T\mathbf{x}_i - \mathbf{W}^T\mathbf{x}_j\right\|^2 m_{ij} = 1$$

Solution

Also converted to a generalized eigen problem by using Laplacian Multiplier.

Linear Discriminant Embedding

Similar to Local Discriminant Embedding with two differences:

(1) Non-zero weights m_{ij} and m_{ij}' are all the same, set to 1.

(2) m_{ij} are non-zero when i, j from the same class, m_{ij}' are non-zero when i,j from different classes. From here we see Linear Discriminant Embedding is a supervised learning method.

(3) Instead of a constrained optimization, it solves an unconstrained problem:

$$\max \frac{\sum_{i,j} \left\|\mathbf{W}^T\mathbf{x}_i - \mathbf{W}^T\mathbf{x}_j\right\|^2 m_{ij}'}{\sum_{i,j} \left\|\mathbf{W}^T\mathbf{x}_i - \mathbf{W}^T\mathbf{x}_j\right\|^2 m_{ij}}$$

Why this method is called “Linear”? As all the others are linear.

Self-Organized Approach

Multidimensional Scaling

Objective

MDS is a set of methods to get a parameterized representation of a data set with only a definition of pair-wise distances, which can be something unlike distance at all, e.g. some kind of dissimilarity measure or some distance measure on non-Euclidean spaces.

General Form

$$\min \sum_{i,j} w_{ij} \left(f\left(\left\|\mathbf{x}_i - \mathbf{x}_j\right\|\right) - g\left(\left\|\mathbf{y}_i - \mathbf{y}_j\right\|\right)\right)^2$$

f and g are some kind of transformation function, w_{ij} could be some kind of weight.

Isomap

Step1. Build neighbor graph using kNN or ϵ-ball.

Step2. Find shortest paths between any two vertices of the graph. Edges weighted by distances on source space.

Step3. Calculate shortest paths and their lengths defined as graph distances $d_G\left(\mathbf{x}_i, \mathbf{x}_j\right)$.

Step4. Apply Multidimensional Scaling on distance matrix $\mathbf{D}_G = \left\{d_{ij} = d_G\left(\mathbf{x}_i, \mathbf{x}_j\right)\right\}$ to solve for the target space coordinates $\mathbf{y}_i$.

Locally Linear Embedding

Assumption

The local patches of source space and target space share the same linear structure.

Derivation

Step1. Build neighbor graph using kNN or ϵ-ball.

Step2. For each i, solve all w_{ij} $\begin{cases} \min \left\| \mathbf{x}_i - \sum_{j \in N(i)} w_{ij} \mathbf{x}_j \right\|^2 \\ s.t. \quad \sum_{j \in N(i)} w_{ij} = 1 \end{cases}$ (Still using Laplacian Multiplier)

Step3. Solve all $\mathbf{y}_i$ by $\begin{cases} \min \sum_{i=1}^{n} \left\| \mathbf{y}_i - \sum_{j \in N(i)} w_{ij} \mathbf{y}_j \right\|^2 \\ s.t. \ \sum_{i=1}^{n} \|\mathbf{y}_i\| = 1 \ \text{and} \ \overline{\mathbf{y}}_i = 0 \end{cases}$

Problem

Require the data points to distribute uniformly in the manifold, otherwise the manifold cannot be extracted effectively.

Laplacian Eigenmaps

Objective

Minimize distances of neighbor vectors using graph Laplacian.

Derivation

Step1. Build neighbor graph using kNN or ϵ-ball.

Step2. Calculate weight matrices

$$\mathbf{M} = \{m_{ij}\}$$

$$m_{ij} = \begin{cases} e^{-\|\mathbf{x}_i - \mathbf{x}_j\|/t}, & \text{if } \langle i, j \rangle \text{ is an edge} \\ 0, & \text{otherwise} \end{cases}$$

Step3. Solve the optimization for the projection matrix $\mathbf{W}$:

$$\min \sum_{i,j} \left\| \mathbf{W}^T \mathbf{x}_i - \mathbf{W}^T \mathbf{x}_j \right\|^2 m_{ij}$$

Solution

Solving for all the $\mathbf{y}_i$ by the following scheme:

$$\mathbf{D} = diag\{m_{ii}\}$$
$$\mathbf{L} = \mathbf{D} - \mathbf{M}$$
$$\mathbf{Y} = \{\mathbf{y}_1 \cdots \mathbf{y}_n\}$$
Solve
$$\mathbf{L} \cdot \mathbf{Y}^T = \lambda \mathbf{D} \cdot \mathbf{Y}^T$$

Why choosing the smallest non-zero eigen values?

Locality Preserving Projection

TODO: This should be categorized into the Linear Projection Approach section.

Objective

Substituting y by x to achieve a new formulation of the problem, which yields a solution with multiple profits:

(1) Computation time reduced from O(n) to O(D)
(2) By solving the projection matrix instead of the target space vectors, LPP is defined everywhere.
(3) Non-linear cases can be handled by using Kernel LPP.

Derivation

Similar to Laplacian Eigenmaps except that

$$\mathbf{D} = diag\left\{m_{ii}\right\}$$
$$\mathbf{L} = \mathbf{D} - \mathbf{M}$$
$$\mathbf{X} = \left\{\mathbf{x}_1 \cdots \mathbf{x}_n\right\}$$

Solve

$$\mathbf{X} \cdot \mathbf{L} \cdot \mathbf{X}^T \cdot \mathbf{e}_j = \lambda \mathbf{X} \cdot \mathbf{D} \cdot \mathbf{X}^T \cdot \mathbf{e}_j$$

Why it is called “Linear” while the Laplacian Eigenmaps is not? As they are all generalized eigen problem.

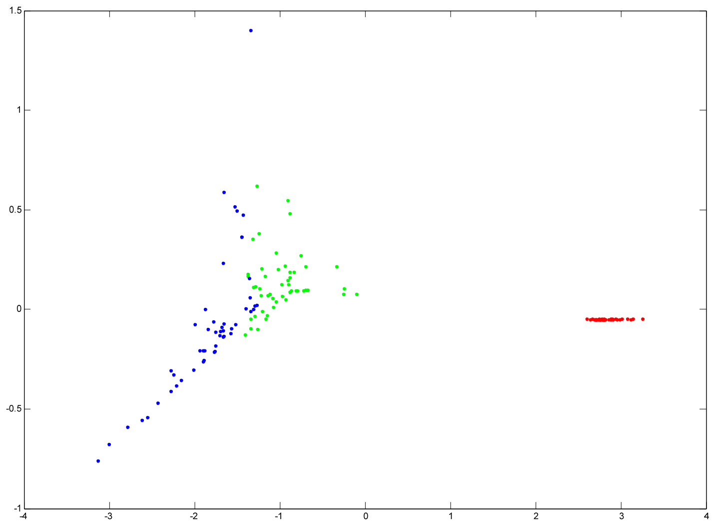
Test Cases

Test Data

See srcData.mat, there are 150 vectors with dimension 4, each is assigned a class id from 1-3 in prior. The data is from Xu Congfu’s course of AI introduction as the data for doing classification tasks.

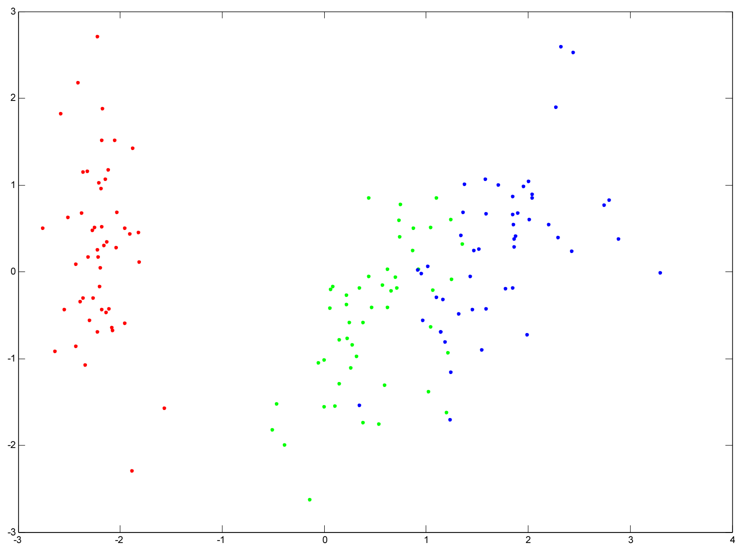
Case 1: ISOMAP

Directly using ISOMAP, kNN Neighbor strategy, a large k with 24 is required to get visually plausible result. Even though, ISOMAP is good enough for the classification task in this example.



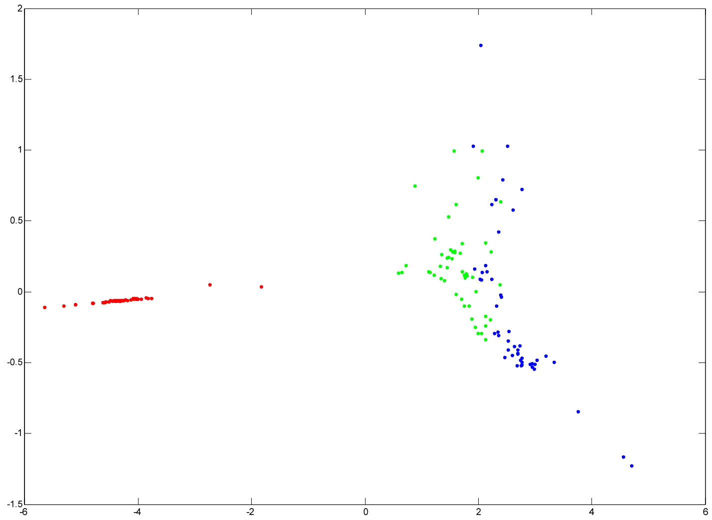
Case 2: PCA

Use PCA only, we see that although PCA seems to be a good representative model, it is not so good for classification, it should be used in combination with other methods.



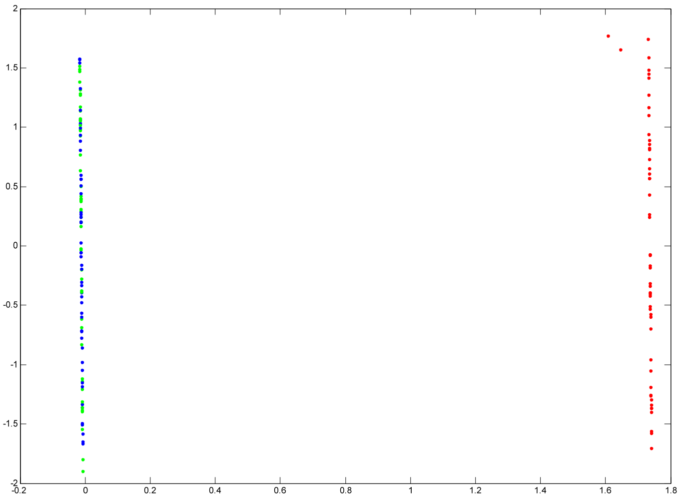
Case 3: ISOMAP on PCA

Used PCA and then ISOMAP, due to the preprocessing by PCA, proceeded ISOMAP can use a less k parameter for kNN, here we use 5.



Case 4: LLE on PCA

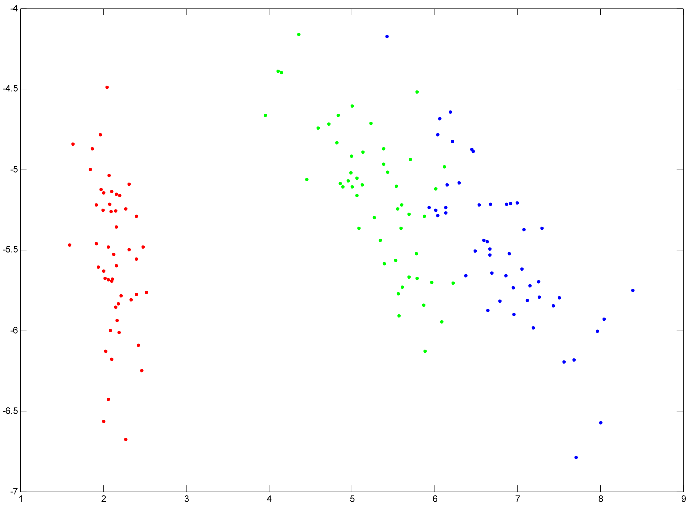
LLE generally fails here, due to the non-uniform distribution of the test data on the manifold. Below is the result after applying PCA and LLE (k=5). Using LLE directly doesn't work at all.



What to do about it ?

Case 5: LPP

It has both the representative power of PCA, while better for classification.



Case 6: LPP on PCA

It seems that the classification property here is not better than simply using LPP