

Dimensionality Reduction and its Application to Semi-supervised Learning

Nandan Marathe

Kanwal Rekhi School of Information Technology,
Indian Institute of Technology, Bombay

Guide: Prof. Sunita Sarawagi
M.Tech Seminar

Introduction

Dimensionality Reduction

- Dimensionality reduction is the mapping of high dimensional data into one with fewer dimensions.
- Formally,
 - Input: $X = (x_1, \dots, x_n)$ with, $x_i \in \mathbb{R}^d$
 - Output: $\psi_i : \psi_i \in \mathbb{R}^m$ such that, $m \ll d$
- When data falls on a smooth, locally flat subspace, then dimensionality reduction is done without any loss of information.

Introduction

Applications

- Effective if the loss due to reduction in dimensionality is less than the gain due to simplifying the problem.
- Can also be viewed as a platform for feature selection, also as a preprocessing platform to obtain better results.
- Other applications include,
 - Image compression
 - Gene expression microarrays
 - Financial time series

Introduction

Classification of Methods

1 Linear Methods

- PCA
- MDS

2 Graphical Methods

- Isomap
- Maximum Variance Unfolding
- Locally Linear Embedding
- Laplacian Eigenmaps

3 Kernel Methods

- Kernel PCA
- Graph based kernels

Principle Component Analysis

- Computes the representation that most faithfully preserves its covariance structure.
- Input patterns projected into the space which minimises the *reconstruction error*

$$\epsilon_{PCA} = \sum_i \left\| \mathbf{x}_i - \sum_{\alpha=1}^m (\mathbf{x}_i \cdot \mathbf{e}_\alpha) \mathbf{e}_\alpha \right\|^2$$

Principle Component Analysis

Steps in PCA

- Centre the data such that the mean lies on origin
- Calculate the covariance Matrix

$$C = \frac{1}{n} \sum_i x_i \cdot x_i^T$$

- Find Eigenvalues and eigenvectors.
- Select the top m eigenvectors as principle components.

Metric Multidimensional Scaling

- Computing lowdimensional representation which best preserves the *inner products*.
- The outputs of MDS minimise the function

$$\epsilon_{MDS} = \sum_{ij} (\mathbf{x}_i \cdot \mathbf{x}_j - \psi_i \cdot \psi_j)^2,$$

- Let $S_{ij} = (\mathbf{x}_i - \mathbf{x}_j)^T (\mathbf{x}_i - \mathbf{x}_j)$
We create a *Gram matrix* $G = -\frac{1}{2}(I - uu^T)S(I - uu^T)$,
where $u = \frac{1}{\sqrt{n}}(1, 1, \dots, 1)^T$.
- If top m eigenvalues of the gram matrix are given by $\{v_\alpha\}_{\alpha=1}^m$ and their respective eigenvalues are $\{\lambda_\alpha\}_{\alpha=1}^m$, the outputs of MDS are given by $\psi_{i\alpha} = \sqrt{\lambda_\alpha} v_{\alpha i}$.

Metric Multidimensional Scaling

- Though based on a different geometric intuition, MDS generates the same outputs as PCA — essentially a rotation + projection.
- In MDS, we approximate the dissimilarity function between input patterns δ_{ij} such that

$$d_{ij} \approx f(\delta_{ij})$$

- Used as a basis for the more complicated non-linear methods.

Graphical Methods

Need for graphical models

- PCA and MDS generate faithful output when the high dimensional input patterns are confined to a low dimensional subspace.
- If input patterns are distributed throughout the subspace, the eigenvalue spectra reveal the data set's high dimensionality - in other words the number of underlying modes of variability.
- In such cases, Graph based methods, emerge as a powerful tool.
- Though capable of revealing highly nonlinear structure, graph-based methods are based on highly tractable (polynomial time) algorithms

Isomap

- Preserves pairwise distances between input patterns, as measured along the submanifold from which they are sampled.
- Euclidean distances substituted by the estimates of geodesic distances along the submanifold.
- Steps of the algorithm:
 - compute the k-nearest neighbours
 - draw edges connecting the k-nearest neighbours
 - assign weights proportional to the Euclidean distance between the neighbours.
 - Compute all the pairwise distances Δ_{ij} between all nodes (i,j) along shortest paths through the graph [$O(n^2 \log n + n^2 k)$]
 - Input these pairwise distances to MDS to get $\psi_i \in \mathbb{R}^m$ for which $\|\psi_i - \psi_j\|^2 \approx \Delta_{ij}^2$

Isomap

- Yields a low dimensional representation in which Euclidean distances between the outputs match the geodesic distances between the input patterns.

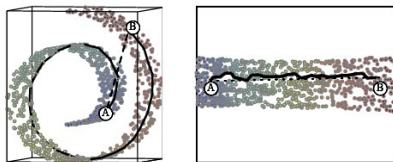


Figure: Isomap applied to a swiss roll

Maximum Variance Unfolding

- Based on preserving the *distances* and *angles* between nearby input patterns.
- Appeals to the notion of isometry to get the low dimensional representation
- Tries to **unfold** a dataset by pulling the input patterns as far apart as possible subject to distance constraints that ensure that the final transformation from input patterns to outputs looks locally like a rotation plus translation.

Maximum Variance Unfolding

Conditions to be met

- A semidefinite programming problem is formulated to solve the following set of equations:

Maximise trace (K) subject to:

$$1) K \succeq 0.$$

$$2) \sum_{ij} K_{ij} = 0.$$

$$3) K_{ij} - 2K_{ji} + K_{jj} = \|x_i - x_j\|^2$$

for all (i, j) such that $\eta_{ij} = 1$ holds

- In the above formulation

$$K_{ij} = \psi_i \cdot \psi_j$$

η is neighbour indicator matrix, such that $\eta_{ij} = 1$ if i and j are nearest neighbours of each other.

Maximum Variance Unfolding

- MVU tries to "unfold" the data set by pulling the input patterns subject to the constraints that ensure that the final transformation looks locally like a rotation plus translation.

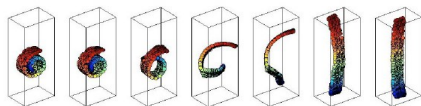


Figure: Maximum variance unfolding applied to a swiss roll

Locally linear embedding

- LLE models the manifold by treating it as a union of linear patches.
- The idea is to express each x_i as a linear combination of its neighbours y_i , and then construct the y_i so that they can be expressed as the same linear combination of its neighbours.
- The algorithm has the following steps:
 - Compute the k -nearest neighbours of each input x_i
 - A directed graph is constructed indicating the nearest neighbour conditions.
 - Weights are assigned to each edge so as to minimise:

$$\epsilon_W = \sum_i \|x_i - \sum_j W_{ij}x_j\|^2$$

where $W_{ij} = 1$ if i and j are nearest neighbours of each other.

Locally linear embedding

- Weights constitute a sparse matrix W that encodes local geometric properties specifying the relation of each point x_i in terms of its nearest neighbours.
- The final outputs are derived which minimise the function:

$$\epsilon_{\psi} = \sum_i \|\psi_i - \sum_j W_{ij} \psi_j\|^2$$

- The d -dimensional embedding that minimises the above function is obtained from the bottom m eigenvectors of the matrix

$$(I - W)^T (I - W)$$

Locally linear embedding

- LLE does not show a tellable gap between the lower m eigenvalues and the other eigenvalues, hence that needs to be passed as a parameter.

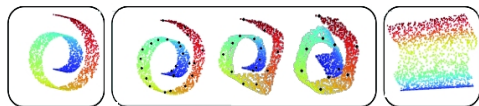


Figure: LLE applied to a swiss roll

Spectral Graph theory

- Data is a graph with weighted, undirected edges such that weights are a measure of similarity between the nodes of the graph.
- Laplacian of a graph:

$$\mathbb{L} \equiv D^{-\frac{1}{2}} L D^{-\frac{1}{2}}$$

where, $L_{ij} \equiv D_{ij} - W_{ij}$ and $D_{ij} \equiv \delta_{ij}(\sum_k W_{ik})$.

- The spectrum of a graph (the eigenvalues and eigenvectors) of \mathbb{L} characterise the properties of a graph.

Laplacian eigenmaps

- Algorithm minimises the cost function:

$$\epsilon_L = \sum_{ij} \frac{W_{ij} \|\psi_i - \psi_j\|^2}{\sqrt{D_{ii} D_{jj}}},$$

where, $W_{ij} = \exp\left(-\frac{\|x_i - x_j\|^2}{2\sigma^2}\right)$ and $D_{ii} = \sum_j W_{ij}$

- Required to calculate the bottom m eigenvectors of

$$\mathbb{L} = I - D^{-\frac{1}{2}} W D^{-\frac{1}{2}}$$

Kernel PCA

Algorithm

- Obtain a kernel matrix K_{ij} such that $K_{ij} = \Phi_i \cdot \Phi_j$ for all i, j .
- Get a **centered** version of the matrix $K^c = PKP$ where, $P \equiv I - \frac{1}{m} e \cdot e'$ where e is a m -vector of all 1's.
- Get eigen vectors of the centered kernel matrix ($\bar{\lambda}$)
- Normalise the corresponding eigenvector, α to have length equal to $\frac{1}{\sqrt{\bar{\lambda}}}$.
- For training point x_k , the principle component is then just $(\Phi(x_k) - \mu) \cdot v = \bar{\lambda} \alpha_k$
- For a general test point x , the principle component is $(\Phi(x) - \mu) \cdot v = \sum_i \alpha_i k(x, x_i) - \frac{1}{m} \sum_{ij} \alpha_i \alpha_j k(x, x_j)$

Kernel PCA

- Can be viewed as yet another method for something like feature selection.
- Removes the disadvantage of classical PCA that it depends only on the first and second moments of data
- Kernel functions can be defined as $k(i, j) = (x_i \cdot x_j + b)^p$ which powers up to $2p$
- Not particularly well suited for manifold learning.




Graph based methods as instances of Kernel PCA

- The gram matrix constructed by Isomap can be viewed as a kernel matrix.
- In maximum variance unfolding, the kernel matrix is learnt by semidefinite programming.
- LLE and Laplacian eigenmaps generate matrices which can be viewed as "inverse" kernel matrices.
- Isomap and MVE have tellable gap between their significant dimensions, which is not seen in LLE and Laplacian; hence the later do not reflect the geometry of the distribution.

Thank You

?

References I

-  L. K. Saul, K. Q. Weinberger, et al. Spectral Methods for Dimensionality Reduction
-  C. J. C. Burges. Geometric methods for feature extraction and dimensional reduction. Data Mining and Knowledge Discovery Handbook: A Complete Guide for Practitioners and Researchers. Kluwer Academic Publishers, 2005.
-  T. Cox and M. Cox. Multidimensional Scaling. Chapman and Hall, London, 1994.