Dimensionality Reduction

Fuying

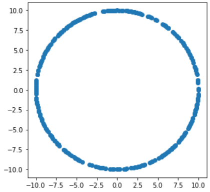
School of Business Administration
South China University of Technology

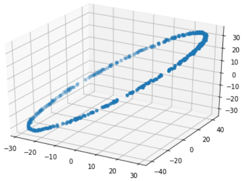
July 26,2019

Outline

Dimensionality reduction:Definition

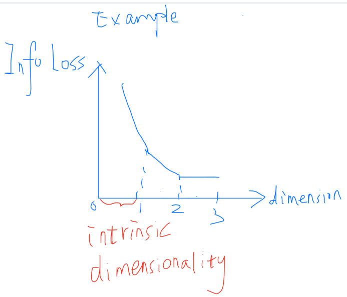
- Dimensionality reduction is the transformation of high-dimensional data into a lower dimensionality(intrinsic dimension exists) to preserve the information as much as possiable or for some specialised tasks (classfication or visualization)
- R=10, $\Theta = (\theta_1, \theta_2, ...\theta_n)$,
  $X_1 = Rsin(\theta), X_2 = Rcos(\theta), Y = A_{(3x2)}X + \epsilon$
- Observed dimension is 3, linear dimension reduction is 2, while intrinsic dimension is 1





Dimensionality reduction:Intrinsic dimensionality

- The ability to reduce redundancy to preserve much information is quiet different for different method.
- Linear methods can only detect the linear redundancy.
- For some task oriented problem, we may redunce the dimensionality into a very low dimension for visualization or classfication even though its intrinsic dimension is larger.



Dimensionality Reduction Goals

1. High dimension leads to the curse of dimensionality $\rightarrow$ data intrinsic dimensionality is lower than observed dimensionality $\rightarrow$ compression of observed data to reduce redundancy to preserve much information as possiable to improve efficiency or reduce overfit.
    1. PCA(Linear method, can only find 2 dimension in last example)
    2. MDS(Linear method)
    3. Kernel PCA(Non-linear method)
    4. Isomap(Manifold learning,Non-linear)
    5. LLE(Manifold learning,Non-linear)
2. For some task oriented problem,we want to reduce dimension for data visualization and exploratory data analysis also need to reduce dimension.
    1. visualization(t-SNE)
    2. classification(LDA)

Preserved information

Preserve info after projecting into lower dimensional space

$Y \in \mathbb{R}^{n \times N} \to \tilde{X} \in \mathbb{R}^{m \times N} \to \tilde{Y} \in \mathbb{R}^{n \times N}, m < n$
s.t. $Info(\tilde{X}) \approx Info(Y)$

- Preserve data points, $\tilde{Y} = f(\tilde{X}) \approx Y$
- Preserve pair wise distance (or dot products)

$$d(\tilde{\mathbf{x}}_\mathbf{i}, \tilde{\mathbf{x}}_\mathbf{j}) \approx d(\mathbf{y}_\mathbf{i}, \mathbf{y}_\mathbf{j}), \forall i, \forall j$$

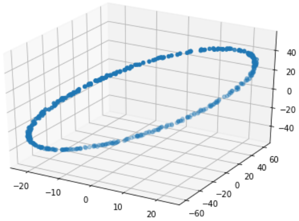
- Preserve local information

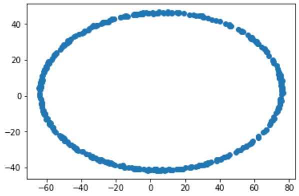
$$d(\tilde{\mathbf{x}}_\mathbf{i}, \tilde{\mathbf{x}}_\mathbf{j}) \approx d(\mathbf{y}_\mathbf{i}, \mathbf{y}_\mathbf{j}), \forall i, \forall j \in \mathrm{Neighbor}(i)$$

- Preserve task-specific information(classfication or visualization)

Preserve data points

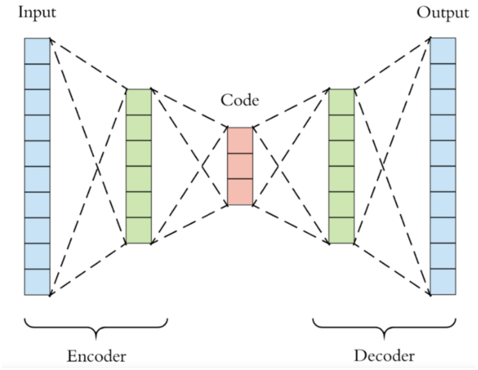
- $Y \in \mathbb{R}^{n \times N} \to \tilde{X} \in \mathbb{R}^{m \times N} \to \tilde{Y} \in \mathbb{R}^{n \times N}, m < n$
- minimize $\left\| \tilde{Y} - Y \right\|_F^2$
- Preserving data points is the most elaborate method.
- Example:PCA and Auto-Encoder





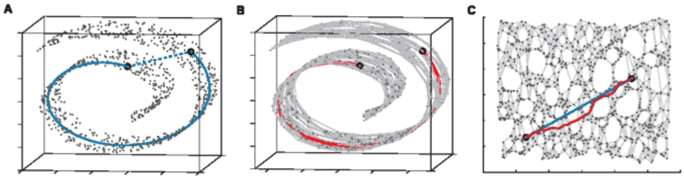
Preserve data points(.cont)

- Comprehensive Introduction to Auto-Encoder
- Auto-Encoder can deal with non-linear problem compared to PCA



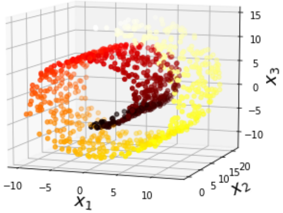
Preserve pairwise distance(or dot products)

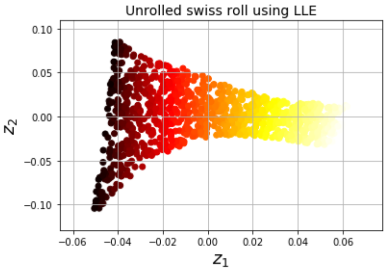
- Gram Matrix: $G_{ij} = \mathbf{y_i} \cdot \mathbf{y_j} = \mathbf{\tilde{x}_i} \cdot \mathbf{\tilde{x}_j}$ and $\tilde{G}_{ij} = \mathbf{\tilde{y}_i} \cdot \mathbf{\tilde{y}_j}$
- $J(Y, \tilde{Y}) = \left\| G - \tilde{G} \right\|_F^2$
- $D_{ij} = \mathbf{x_i} \cdot \mathbf{x_i} + \mathbf{x_j} \cdot \mathbf{x_j} - 2\mathbf{x_i} \cdot \mathbf{x_j} = G_{ii} + G_{jj} - 2G_{ij}$
- One-one-correspondence between D and G
- When the distance is Euclidean distance(PCA equals to MDS)
- When the distance is Geodesic distance(Isomap)



Preserve local information

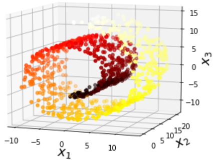
- LLE is a local method compared with MDS or Isomap as a global method
- $d(\mathbf{x_i}, \mathbf{x_j}) \approx d(\mathbf{y_i}, \mathbf{y_j}), \forall i, \forall j \in \text{Neighbor}(i)$
- Each instance can be linearly represented by its several nearby instances.

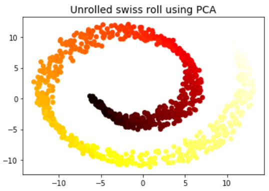


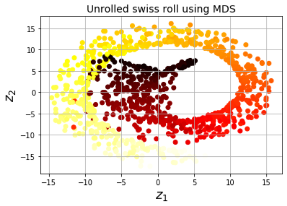


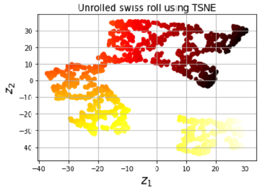
Intuitive view of some methods

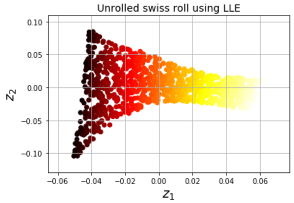
- Manifold (an example of Swiss roll):

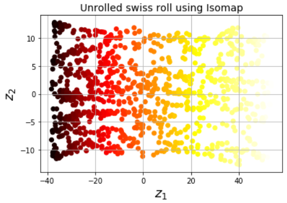






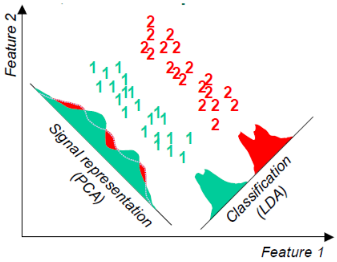


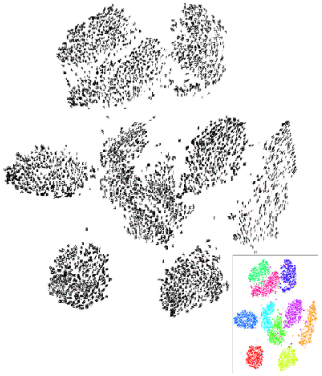




Preserve task-specific information

- Task oriented: classfication and visualization(MINIST dataset)
- The dimensionality after being reduced may lower than its intrinsic dimensionaity.





Outline

Dimensionality Reduction Goals in PCA:Preserve data points

- Goal:Reduce redundancy in the data and preserve as much information as possiable
- Intrinsic Variances $X \in \mathbb{R}^{m \times N}$,uncorrected
- Observed Variances $Y \in \mathbb{R}^{n \times N}$ are given by $Y = AX$
- Assume $n > m$
- Find U to finish a tranformation(U is orithonormal): $\tilde{X} = U^T Y$ and Reconstract $\tilde{Y} = U\tilde{X}$
- Minimize $\left\| Y - \tilde{Y} \right\|_F^2$
- Equivalent to the maximum variance

PCA:Equivalent to the maximum variance

- $\mathbf{y_i}$ as the i^th column of Y,$i = 1, 2, ...N$,supposed to be zero-mean
- Firstly, we just want to find a single direction.
- Let $\mathbf{u_1}$ denote the principle axis(u is orthonormal)
- We want to maximize the following function:

$$\frac{1}{N}\sum_{i=1}^{N}(\mathbf{y_i}\cdot\mathbf{u_1})^2 = \frac{1}{N}\sum_{i=1}^{N}(\mathbf{y_i^T u_1})^{\mathbf{2}}$$

- Which is

$$\frac{1}{N}\sum_{i=1}^{N}(\mathbf{y_i^T u_1})^2 = \frac{1}{N}\sum_{i=1}^{N}\mathbf{u_1^T y_i y_i^T u_1} = \mathbf{u_1}^T\frac{1}{N}YY^T\mathbf{u_1}$$

PCA:Equivalent to the maximum variance among data points after projection

- $\mathbf{u_1} = \arg\max_u \mathbf{u^T} \frac{1}{N} YY^T \mathbf{u}$ (s.t. $\mathbf{u^T u} = 1$)
- The corresponding Lagrangian is given by:

$$L(\mathbf{u}, \lambda) = \mathbf{u^T} \frac{1}{N} YY^T \mathbf{u} - \lambda(\mathbf{u^T u} - 1)$$

- Taking the gradient and setting it equal to zero we get:

$$\frac{1}{N} YY^T \mathbf{u} = \lambda \mathbf{u}$$

- $\frac{1}{N} YY^T$is symmetric and positive semi-definite matrix
- Covariance matrix $\frac{1}{N} YY^T$ has an eigen-decomposition

$$\frac{1}{N} YY^T = U \Lambda_{PCA} U^T$$

PCA Algorithm

- Inputs: Observed data matrix Y and number of PCA modes k
- output: Recovered intrinsic variables $\tilde{X}$ and reconstructed $\tilde{Y}$
- step1: Compute the mean $\mu_j = \frac{1}{N} \sum_{i=1}^{N} Y_{ji}$
- step2: Center the data: Subtract μ form each column of Y
- step3:Compute the eigen-decomposition of $\frac{1}{N} YY^T$:

$$\frac{1}{N} YY^T = U \Lambda_{PCA} U^T$$

- step4:Select the top k eigen vectors U=U(:,1:K)
- step5: Project onto the principal components $\tilde{X} = U^T Y$
- step6:Reconstract $\tilde{Y} = U\tilde{X} + \mu$

Outline

MDS:Preserve pairwise Euclidean distance(or dot product)

- MDS preserves the pairwise distance rather than data points compared to PCA or Auto-Encoder.
- We only know the distance(e.g.Euclidean) of each original point in dimension n,Which can be defined as D
- Let $\mathbf{x}(\tilde{\mathbf{i}}) \in R^m$ be the i^{th} column of $\tilde{X}$
- For any two points i and j: $D_{ij} = \|\mathbf{y_i} - \mathbf{y_j}\|_F^2 = \|\tilde{\mathbf{x_i}} - \tilde{\mathbf{x_j}}\|_F^2$

MDS:Preserve pairwise Euclidean distance(or dot product)

- Given that: $Y = A\tilde{X}$
- Instead of corrleations,define a Gram matrix:$G = Y^T Y$
- Suppose that A is orthogonal(if not, we can do a transmision to make A is orthogonal): $G = Y^T Y = \tilde{X}^T A^T A\tilde{X} = \tilde{X}^T\tilde{X}$
- G is an NxN symmetric positive semi-definite matrix, which is the pairwise inner products of the data points,that is: $G_{ij} = (\tilde{X}^T\tilde{X}) = \tilde{x}_i \cdot \tilde{x}_j$

MDS Algorithm

- Input:Given distance matrix **D**
- $D_{ij} = \|\mathbf{y_i} - \mathbf{y_j}\|_F^2 = \|\tilde{\mathbf{x}}_\mathbf{i} - \tilde{\mathbf{x}}_\mathbf{j}\|_F^2 = \|\tilde{\mathbf{x}}_\mathbf{i} \cdot \tilde{\mathbf{x}}_\mathbf{i}\| + \|\tilde{\mathbf{x}}_\mathbf{j} \cdot \tilde{\mathbf{x}}_\mathbf{j}\| - 2\,\|\tilde{\mathbf{x}}_\mathbf{i} \cdot \tilde{\mathbf{x}}_\mathbf{j}\|$
- By constructing Gram matrix G,we know that:
  $G = Y^TY = \tilde{X}^TA^TA\tilde{X} = \tilde{X}^T\tilde{X}$
- Output:$\tilde{X}$
- So we can calculate X by:
  $D \to G = Y^TY \to G = Y^TY = \tilde{X}^T\tilde{X}$
- Question: What is the relationship between D and G?

One-one-correspondence between D and G

- $D_{ij} = \|\tilde{\mathbf{x}}_\mathbf{i} \cdot \tilde{\mathbf{x}}_\mathbf{i}\| + \|\tilde{\mathbf{x}}_\mathbf{j} \cdot \tilde{\mathbf{x}}_\mathbf{j}\| - 2\,\|\tilde{\mathbf{x}}_\mathbf{i} \cdot \tilde{\mathbf{x}}_\mathbf{j}\|$
- Centering X: $\frac{1}{N}\sum_i \tilde{\mathbf{x}}_\mathbf{i} = 0$ and $\frac{1}{N}\sum_i \tilde{\mathbf{x}}_\mathbf{i} \cdot \tilde{\mathbf{x}}_\mathbf{i} = \sigma^2$
  $$\frac{1}{N}\sum_i D_{ij} = \frac{1}{N}\sum_j(\tilde{\mathbf{x}}_\mathbf{i} \cdot \tilde{\mathbf{x}}_\mathbf{i} + \tilde{\mathbf{x}}_\mathbf{j} \cdot \tilde{\mathbf{x}}_\mathbf{j} - 2\tilde{\mathbf{x}}_\mathbf{i} \cdot \tilde{\mathbf{x}}_\mathbf{j} = \sigma^2 + \tilde{\mathbf{x}}_\mathbf{j} \cdot \tilde{\mathbf{x}}_\mathbf{j})$$
  $$\frac{1}{N}\sum_j D_{ij} = \frac{1}{N}\sum_j(\tilde{\mathbf{x}}_\mathbf{i} \cdot \tilde{\mathbf{x}}_\mathbf{i} + \tilde{\mathbf{x}}_\mathbf{j} \cdot \tilde{\mathbf{x}}_\mathbf{j} - 2\tilde{\mathbf{x}}_\mathbf{i} \cdot \tilde{\mathbf{x}}_\mathbf{j} = \sigma^2 + \tilde{\mathbf{x}}_\mathbf{i} \cdot \tilde{\mathbf{x}}_\mathbf{i})$$
  $$\frac{1}{N^2}\sum_{i,j} D_{ij} = \frac{1}{N}\sum_j(\frac{1}{N}\sum_i D_ij) = \frac{1}{N}\sum_j(\sigma^2 + \tilde{\mathbf{x}}_\mathbf{j} \cdot \tilde{\mathbf{x}}_\mathbf{j}) = 2\sigma^2$$
- $\mathbf{x}_\mathbf{i} \cdot \mathbf{x}_\mathbf{j} = G_{ij} = -\frac{1}{2}(D_{ij} - \frac{1}{N}\sum_i D_{ij} - \frac{1}{N}\sum_j D_{ij} + \frac{1}{N^2}\sum_{i,j} D_{ij})$
- Let $\mathbb{I}$ be the NxN matrix with all the element is 1,then:
  $$G = -\frac{1}{2}(D - \mathbb{I}D\frac{1}{N} - \frac{1}{N}D\mathbb{I} + \frac{1}{N}D\frac{1}{N})$$

MDS Algorithm

- One-one-correspondence between D and G
- $G = Y^T Y = \tilde{X}^T A^T A \tilde{X} = \tilde{X}^T \tilde{X}$
- Compute the eigen-decomposition of $G = V \Lambda_M DSV^T$
- Dimensionality Reduction:Set $\widetilde{X} = I_{pN} \Lambda_M DSV^T$
- $\widetilde{X}$ are the p-dimensional corrdinates with the cloest Gram matrix to X,minimizes the residual R:
  $G = X^T X = \widetilde{X}^T \widetilde{X} + \sum_{j=p+1}^{N} (\lambda_{MDS})_j v(j) v(j)^T = \widetilde{G} + R$

Equivalence of MDS and PCA

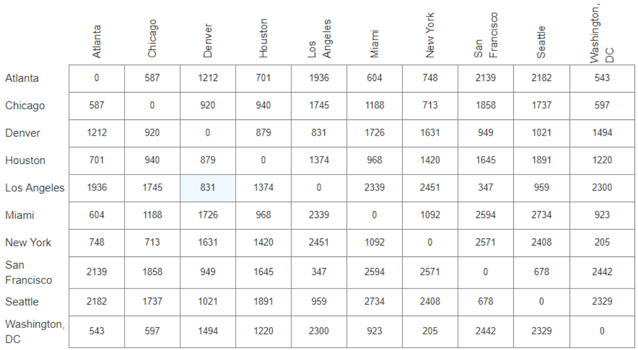
- PCA needs Y to compute correlations: $\frac{1}{N}YY^T = U_{PCA}\Lambda_{PCA}U_{PCA}^T$,then $X_{PCA} = I_{PxN}U_{PCA}^T Y$
- MDS needs Y to compute Gram matrix: $D \rightarrow G = \tilde{X}^T\tilde{X} = Y^TY = V_{MDS}\Lambda_{MDS}V_{MDS}^T$,then $X_{MDS} = I_{PxN}\Lambda_{MDS}^{0.5}V_{MDS}^T$
- To prove the equivalence of MDS and PCA , Suppose that Y has singular value decomposition:$Y = USV^T$(U or V is orthonormal,Respectively)

Equivalence of MDS and PCA(cont.)

$$\frac{1}{N} YY^T = \frac{1}{N}(USV^T)(VS^TU^T)$$
$$= U\frac{SS^T}{N}U^T$$
$$= U_{PCA}\Lambda_{PCA}U_{PCA}^T$$
$$Y^TY = (VS^TU^T)(USV^T)$$
$$= VS^TSV^T$$
$$= V_{MDS}\Lambda_{MDS}V_{MDS}^T$$
$$S = \Lambda_{MDS}^{0.5}$$
$$X_{PCA} = I_{p\times N}U_{PCA}^TY$$
$$= I_{p\times N}U^TUSV^T = I_{p\times N}\Lambda_{MDS}^{0.5}V^T$$
$$= I_{p\times N}\Lambda_{MDS}^{0.5}V_{MDS}^T$$
$$= X_{MDS}$$

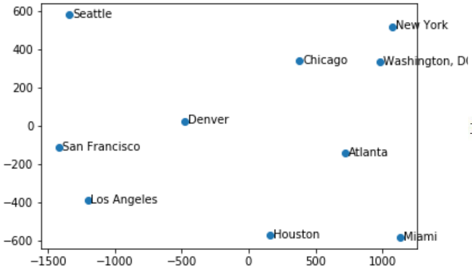
MDS Example

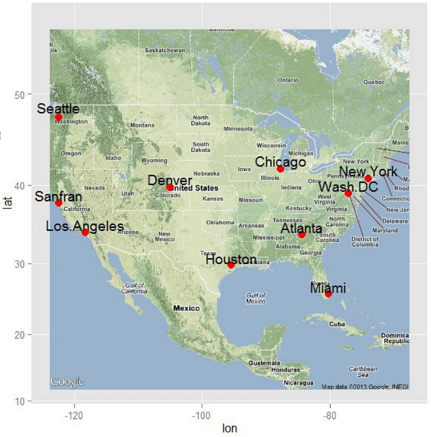
- Table below shows the distances between US cities
- ▸ Data Visualization using Multidimensional Scaling



	Atlanta	Chicago	Denver	Houston	Los Angeles	Miami	New York	San Francisco	Seattle	Washington, DC
Atlanta	0	587	1212	701	1936	604	748	2139	2182	543
Chicago	587	0	920	940	1745	1188	713	1858	1737	597
Denver	1212	920	0	879	831	1726	1631	949	1021	1494
Houston	701	940	879	0	1374	968	1420	1645	1891	1220
Los Angeles	1936	1745	831	1374	0	2339	2451	347	959	2300
Miami	604	1188	1726	968	2339	0	1092	2594	2734	923
New York	748	713	1631	1420	2451	1092	0	2571	2408	205
San Francisco	2139	1858	949	1645	347	2594	2571	0	678	2442
Seattle	2182	1737	1021	1891	959	2734	2408	678	0	2329
Washington, DC	543	597	1494	1220	2300	923	205	2442	2329	0

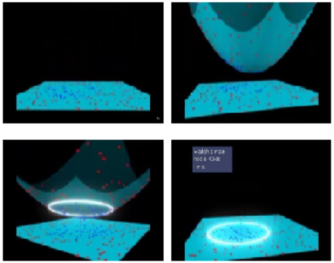
MDS Example

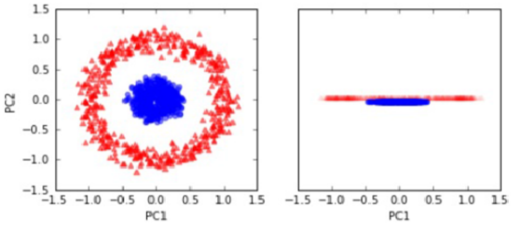

Seattle
New York
Chicago
Washington, DC
Denver
San Francisco
Atlanta
Los Angeles
Houston
Miami
600
400
200
0
−200
−400
−600
−1500
−1000
−500
0
500
1000


Seattle
Sanfran
Los.Angeles
Denver
Chicago
New.York
Wash.DC
Atlanta
Houston
Miami
lat
lon
50
40
30
20
10
-120
-100
-80
Google
Map data ©2013 Google, INEGI

Kernel PCA

- A linear decision boundary in the high-dimensional feature space corresponds to a complex nonlinear decision boundary in the original space.
- It turns out that the same trick can be applied to PCA, making it possible to perform complex nonlinear projections for dimensionality reduction.
- ▸ Watching vedio





Kernel-PCA

- Map **y** in $\mathbb{R}^n$ into a high(possible infinite) dimensional reducing kernel Hibert space

$$y \in \mathbb{R}^n \to \phi(y) \in \mathbb{H}$$

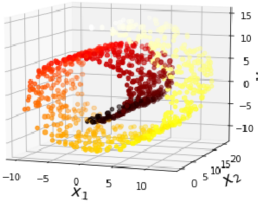
- Kernel $J(y_i, y_j)$ is inner product:

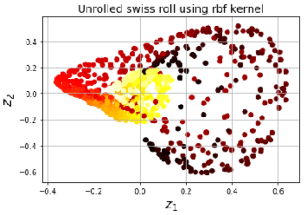
$$J(y_i, y_j) =< \Phi(y_i), \Phi(y_j) >$$

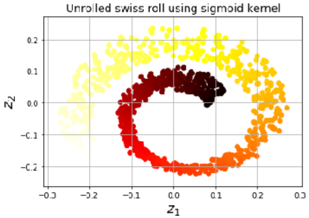
- After mapping into $\mathbb{H}$ and assuming centered data, if J is symmetric and positive definite, it is the same with MDS(Cause gram matrix is also dot product)
- Eigenvectors of matrix J give new coordinates for the data (MDS)

Kernel-PCA for manifold learning will fail

- The swiss roll, reduced to two dimensions using an RBF kernel and a sigmoid kernel, respectively. (Logistic).
- It is often good at preserving clusters of instances after projection.



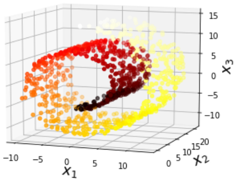


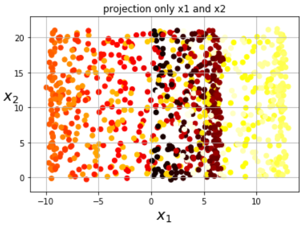


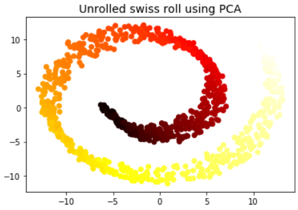
Outline

Projection may fail

- Projection(e.g.PCA) is not always the best approach for dimensionality reduction. In many cases, the subspace may twist and turn, such as in the famous Swiss roll toy dataset.
- In this case, projection method may fail.
- However, what we really want is to unroll the Swiss roll to obtain the 2D dataset.

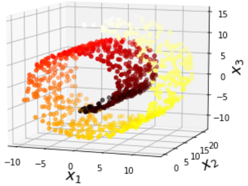


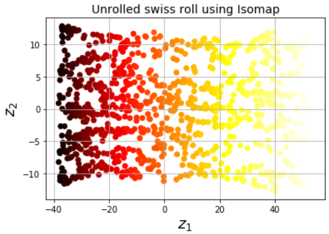




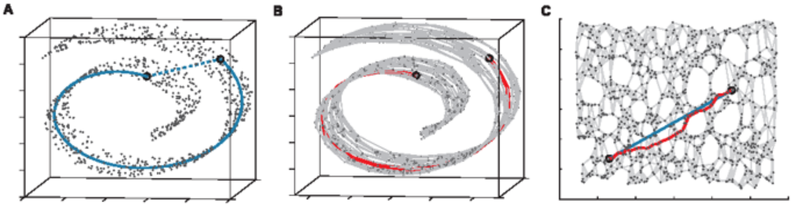
Manifold learning

- Manifold: m-dimensional manifold is a part of an n-dimensional space ($m < n$) that locally resembles a m-dimensional hyperplane(In the case of the Swiss roll, m = 2 and n = 3. it locally resembles a 2D plane, but it is rolled in the third dimension.
- Manifold learning: Modeling the manifold on which the training instances lie.





Isomap:Preserve the global pairwise geodesic distance

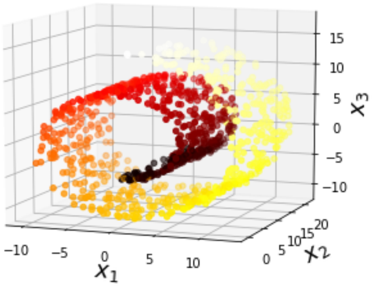


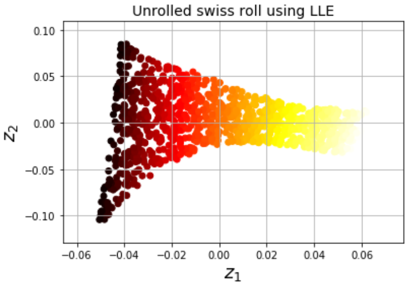
- Identify neighbourhoods around each point (local points, assumed to be local on the manifold). Euclidean distances are preserved within a neighbourhood.
- For points outside the neighbourhood, estimate geodesic distances by shortest path in graph.
- Use classical MDS with geodesic distances(replace Euclidean distance to Geodesic distance)

LLE:Preserve the local pairwise distance

1. A very powerful nonlinear dimensionality reduction technique.
2. LLE works by first measuring how each training instance linearly relates to its closest neighbors.
3. Looking for a low-dimensional representation of the training set where these local relationships are best preserved.
4. particularly good at unrolling twisted manifolds, especially when there is not too much noise.
5. LLE preserves the local pairwise distance while Isomap and MDS preserves the global pairwise distance.

LLE





- Distances between instances are locally well preserved.
- The right part of the unrolled Swiss roll is squeezed, while the left part is stretched.

LLE step1:linearly modeling local relationships

$$\hat{W} = \arg\min_{W} \sum_{i=1}^{m} \|\mathbf{y^{(i)}} - \sum_{j=1}^{m} \hat{w}_{i,j}\mathbf{y^{(j)}}\|$$

$$\text{subject to} \begin{cases} w_{i,j} = 0 & \text{if } y^{(j)} \text{ is not one of the k c.n. of } \mathbf{y^{(i)}} \\ \sum_{i=1}^{m} w_{i,j} = 1 & \text{for } i = 1, 2, ..., m \end{cases}$$

- Each instance is linearly represented by its several nearby instances.
- W encodes the local linear relationships between the training instance.

LLE step2:reduce dimensionality while preserving relationships

$$\tilde{X} = \arg\min_{\tilde{x}} \sum_{i=1}^{m} \|\tilde{\mathbf{x}}^{(\mathbf{i})} - \sum_{j=1}^{m} \hat{w}_{i,j}\tilde{\mathbf{x}}^{(\mathbf{j})}\|$$

- Keeping the weight fixed and finding the optimal position of the instances' images in the low-dimensional space.
- By solving the above optimization problem can obtain the mapped data set $\tilde{X}$.
- A regression learner can be trained to predict the low-dimensional space coordinates of the new sample.

Summary

- Preserve each individual data point
  - $Y \rightarrow \tilde{X} \rightarrow \tilde{Y}$
  - To minimize $J(Y, \tilde{Y}) = \sum_{i=1}^{N} \|\mathbf{y_i} - \mathbf{\tilde{y}_i}\|$
  - Example:Auto Encoder
- Preserve pairwise distance(or dot product)
  - $G_{ij} = \mathbf{\tilde{x}_i} \cdot \mathbf{\tilde{x}_j} = \mathbf{y_i} \cdot \mathbf{y_j}$ and $\tilde{G}_{ij} = \mathbf{\tilde{y}_i}\mathbf{\tilde{y}_j}$
  - $D_{ij} = \mathbf{\tilde{x}_i} \cdot \mathbf{\tilde{x}_i} + \mathbf{\tilde{x}_j} \cdot \mathbf{x_j} - \mathbf{2}\mathbf{\tilde{x}_i} \cdot \mathbf{\tilde{x}_j}$
  - $J(Y, \tilde{Y}) = \left\| G - \tilde{G} \right\|_2$
  - Euclidean distance(PCA);Geodesic distance(Isomap)
- Preserve local information
  - LLE(linearly represented by its several nearby instance)
- Task oriented
  - LDA(classfication)
  - t-SNE(visualization)

Reference

- http://math.gmu.edu/ berry/Presentations/PCAMDS.pdf(PCA and MDS)
- manifolds/lecture13.ppt(PCA and MDS)
- Books: Machine Learning: A Bayesian and Optimization Perspective(PCA and Kernal PCA in Chapter 19)
- Books: Hands-On Machine Learning with Scikit-Learn and TensorFlow(Manifold learning in chapter 8)

More

$$S = E((x-\mu)(x-\mu)^T)$$

$$S^T = S$$

so S is symmetric

$$uSu^T = uE((x-\mu)(x-\mu)^T)u^T$$

$$= E(u(x-\mu)(x-\mu)^T u^T)$$

$$= E(u(x-\mu)(u(x-\mu))^T)$$

$$= E(u(x-\mu)^2) \geqslant 0$$

so S is semi-definite.