Introduction to Machine Learning

Lecture 17: Dimensionality Reduction

Principal Component Analysis (PCA)

Prof. Subrahmanya Swamy Peruru
Department of Electrical Engineering
IIT Kanpur

Dimensionality Reduction: Example

Consider 10 data samples in $\mathbb{R}^4$: (Unlabelled data)

$$\mathbf{x}_1 = (1,1,2,2)^{\mathrm{T}},\ \mathbf{x}_2 = (-1,-1,3,3)^{T},\ \mathbf{x}_3 = (2,2,-2,-2)^{T},\ \mathbf{x}_4 = (-3,-3,1,1)^{\mathrm{T}},\ \mathbf{x}_5 = (4,4,0,0)^{\mathrm{T}},$$

$$\mathbf{x}_6 = (0,0,4,4)^{\mathrm{T}},\ \mathbf{x}_7 = (-2,-2,-1,-1)^{\mathrm{T}},\ \mathbf{x}_8 = (3,3,-3,-3)^{\mathrm{T}},\ \mathbf{x}_9 = (-4,-4,2,2)^{\mathrm{T}},\ \mathbf{x}_{10} = (1,1,-4,-4)^{\mathrm{T}}.$$

Hidden Structure in data: Any vector in our data is of the form $(a,a,b,b)^{\mathrm{T}}$.

Idea: Choose two basis directions in $\mathbb{R}^4$ and express any vector in data as

$$\mathbf{w}_1 = \begin{bmatrix}1\\1\\0\\0\end{bmatrix}, \qquad \mathbf{w}_2 = \begin{bmatrix}0\\0\\1\\1\end{bmatrix}, \qquad \mathbf{x} = a\begin{bmatrix}1\\1\\0\\0\end{bmatrix} + b\begin{bmatrix}0\\0\\1\\1\end{bmatrix} = \begin{bmatrix}a\\a\\b\\b\end{bmatrix}.$$

Low dimensional representation z:

$$\mathbf{x} = (a,a,b,b)^{T} \Leftrightarrow\ \mathbf{z} = (a,b)^{T}$$

Reduced Storage: Original data – 4×10, Low dimensional form – 2×10 ($\mathbf{z}$ vectors) $+ 4 \times 2$ (basis $\mathbf{w_1}$ and $\mathbf{w_2}$)

Relation:

$$\mathbf{x} = \mathrm{z}_1\,\mathbf{w}_1 + \mathrm{z}_2\,\mathbf{w}_2$$

In practice, exact reconstruction is not possible.

Dimensionality Reduction: Example

Each sample is a 64×64 pixel image

$\mathbf{x} \in \mathbb{R}^{4096}$

Fig Source: PML book by Murphy

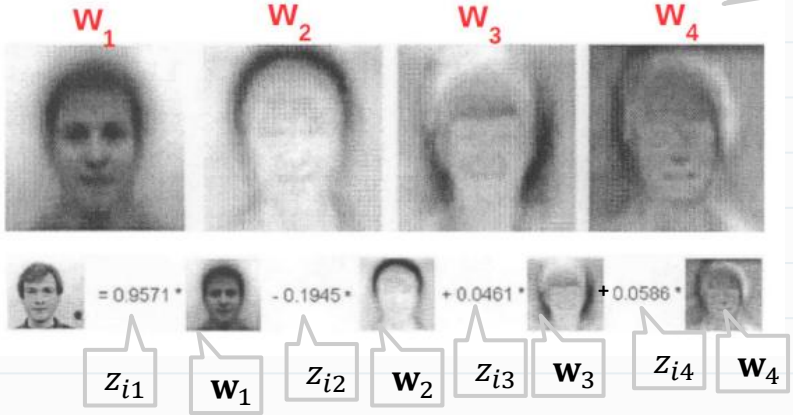
Dimensionality Reduction: Example

Dim-red for face images

$$\mathbf{x_i} \approx z_{i1}\,\mathbf{w}_1 + z_{i2}\,\mathbf{w}_2 + z_{i3}\,\mathbf{w}_3 + z_{i4}\,\mathbf{w}_4$$

K=4 "basis" face images

Each "basis" image is like a "template" that captures the common properties of face images in the dataset



A face image $\mathbf{x}_i \in \mathbb{R}^{4096}$

In this example, $\boldsymbol{z}_i \in \mathbb{R}^K$ $(K = 4)$ is a low-dim feature rep. for $\mathbf{x}_i \in \mathbb{R}^d$ $(d = 4096)$

Essentially, each face image in the dataset now represented by just 4 real numbers

Slide adapted from Prof Piyush Rai

Principal Component Analysis (PCA)

Suppose, we want a K −dimensional representation of our data $\mathbf{x}_i \in \mathbb{R}^d, i = 1,2, \dots, n$

$$K < d$$

Approximate representation:

$$\mathbf{x} \approx \hat{\mathbf{x}} = z_1 \mathbf{w}_1 + z_2 \mathbf{w}_2 + z_3 \mathbf{w}_3 + \cdots + z_K \mathbf{w}_K$$

Idea of PCA: Find a set of orthonormal basis vectors (projection directions) $\mathbf{w}_1, \mathbf{w}_2, \dots, \mathbf{w}_K$ that minimize the reconstruction error

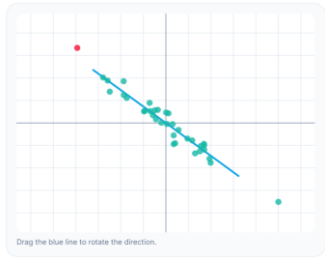
Why are we calling the basis vectors as projection directions?

$$\sum_{i=1}^{n} \| \mathbf{x}_i - \hat{\mathbf{x}}_i \|_2^2$$

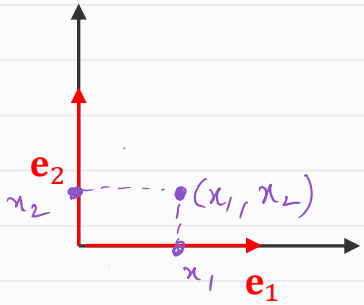
Fact:

The K orthonormal directions that minimize the total reconstruction error are exactly the directions that maximize the variance of the projected points.

PCA with $K = 1$



Demo Link: https://numiqo.com/lab/pca



$$\begin{bmatrix} x_1 \\ x_2 \end{bmatrix} = x_1 \begin{bmatrix} 1 \\ 0 \end{bmatrix} + x_2 \begin{bmatrix} 0 \\ 1 \end{bmatrix}$$

Principal Component Analysis: 1D

Data: n samples in $\mathbb{R}^d$, centered (mean is zero).

$$\mathbf{x}_1, \ldots, \mathbf{x}_n \in \mathbb{R}^d, \qquad \frac{1}{n}\sum_{i=1}^{n} \mathbf{x}_i = \mathbf{0}.$$

Stack samples into the **data matrix**:

$$\mathbf{X} = \begin{bmatrix} -- & -\mathbf{x}_1^\top & -- \\ -- & -\mathbf{x}_2^\top & -- \\ & \vdots & \\ -- & -\mathbf{x}_n^\top & -- \end{bmatrix} \in \mathbb{R}^{n \times d}.$$

Goal (PCA to 1D): choose a unit direction $\mathbf{w} \in \mathbb{R}^d$ maximizing variance of data after projection onto $\mathbf{w}$.

For each sample $\mathbf{x}_i$:

(scalar projection component) $z_i = \mathbf{w}^\top \mathbf{x}_i \in \mathbb{R}$.

1D representation of all the samples:

$$\mathbf{z} = \begin{bmatrix} z_1 \\ z_2 \\ \vdots \\ z_n \end{bmatrix} = \begin{bmatrix} \mathbf{w}^\top \mathbf{x}_1 \\ \mathbf{w}^\top \mathbf{x}_2 \\ \vdots \\ \mathbf{w}^\top \mathbf{x}_n \end{bmatrix} = \begin{bmatrix} -- & -\mathbf{x}_1^\top & -- \\ -- & -\mathbf{x}_2^\top & -- \\ & \vdots & \\ -- & -\mathbf{x}_n^\top & -- \end{bmatrix} \begin{bmatrix} w_1 \\ w_2 \\ \vdots \\ w_d \end{bmatrix} = \mathbf{X}\mathbf{w} \in \mathbb{R}^n.$$

Variance in the projected 1D coordinates

Since the original data are centered, the projected scalars are also centered:

$$\mu_z = \frac{1}{n}\sum_{i=1}^{n} z_i = \frac{1}{n}\sum_{i=1}^{n} \mathbf{w}^\top \mathbf{x}_i = \mathbf{w}^\top \left(\frac{1}{n}\sum_{i=1}^{n} \mathbf{x}_i\right) = \mathbf{w}^\top \mathbf{0} = 0.$$

So the (sample) variance of the 1D projected data is

$$\mathrm{Var}(z) = \frac{1}{n}\sum_{i=1}^{n} (z_i - \mu_z)^2 = \frac{1}{n}\sum_{i=1}^{n} z_i^2 = \frac{1}{n} \| \mathbf{z} \|_2^2.$$

Using $\mathbf{z} = \mathbf{X}\mathbf{w}$:

$$\mathrm{Var}(z) = \frac{1}{n} \| \mathbf{z} \|_2^2 = \frac{1}{n} \| \mathbf{X}\mathbf{w} \|_2^2.$$

PCA-1D objective: Choose a unit direction $\mathbf{w} \in \mathbb{R}^d$ maximizing variance of projected data $\mathrm{Var}(z)$.

$$\max_{\|\mathbf{w}\|_2 = 1} \frac{1}{n} \| \mathbf{X}\mathbf{w} \|_2^2$$

Solving using SVD

PCA-1D objective:

$$\max_{\|\mathbf{w}\|_2=1} \frac{1}{n} \| \mathbf{Xw} \|_2^2$$

Since $\frac{1}{n} > 0$ is a constant independent of $\mathbf{w}$, it does not affect the maximiser.

Also, since the square function is strictly increasing on $[0, \infty)$, removing the square does not change the maximiser.

$$\max_{\|\mathbf{w}\|_2=1} \frac{1}{n} \| \mathbf{Xw} \|_2^2 = \max_{\|\mathbf{w}\|_2=1} \| \mathbf{Xw} \|_2^2 = \max_{\|\mathbf{w}\|_2=1} \| \mathbf{Xw} \|_2$$

$\max_{\|\mathbf{w}\|_2=1} \| \mathbf{Xw} \|_2$ is about finding the direction of the maximum stretching.

Recall that the first right singular vector $\mathbf{v_1}$ (corresponding to the largest singular value σ_1) is the direction of maximum stretching.

Hence, variance of the projected data is maximum when $\mathbf{w}$ aligns with the top right singular−direction of data matrix $\mathbf{X}$.

Solving using EVD

PCA-1D objective: $\frac{1}{n} \| \mathbf{X}\mathbf{w} \|_2^2$

Expand as a quadratic form:

$$\frac{1}{n} \| \mathbf{X}\mathbf{w} \|_2^2 = \frac{1}{n} (\mathbf{X}\mathbf{w})^\top (\mathbf{X}\mathbf{w}) = \mathbf{w}^\top \left(\frac{1}{n} \mathbf{X}^\top \mathbf{X} \right) \mathbf{w} = \mathbf{w}^\top \mathbf{S}\, \mathbf{w}.$$

Thus, PCA-1D objective in terms of covariance matrix $\mathbf{S}$**:**

$$\max_{\|\mathbf{w}\|_2 = 1} \mathbf{w}^\top \mathbf{S}\, \mathbf{w}.$$

Since $\mathbf{S}$ is symmetric, EVD gives: $\mathbf{S} = \mathbf{Q}\mathbf{\Lambda}\mathbf{Q}^\top$, in terms of the orthonormal eigenvectors $\mathbf{q}_i$ and eigen values

$$\mathbf{Q} = [\mathbf{q}_1 \quad \mathbf{q}_2 \quad \cdots \quad \mathbf{q}_d], \qquad \mathbf{\Lambda} = \mathrm{diag}(\lambda_1, \ldots, \lambda_d), \quad \lambda_1 \geq \cdots \geq \lambda_d \geq 0.$$

Substitute into objective:

$$\mathbf{w}^\top \mathbf{S} \mathbf{w} = \mathbf{w}^\top (\mathbf{Q}\mathbf{\Lambda}\mathbf{Q}^\top) \mathbf{w}.$$

Let $\mathbf{u} = \mathbf{Q}^\top \mathbf{w}$. Since $\mathbf{Q}$ is orthogonal: $\| \mathbf{u} \|_2 = \| \mathbf{w} \|_2$.

Then

$$\mathbf{w}^\top \mathbf{S} \mathbf{w} = \mathbf{u}^\top \mathbf{\Lambda} \mathbf{u}.$$

Solving using EVD (Contd.)

Now solve:

$$\max_{\|\mathbf{u}\|_2=1} \mathbf{u}^\top \mathbf{\Lambda} \mathbf{u}.$$

Since $\mathbf{\Lambda}$ is diagonal:

$$\max\ \mathbf{u}^\top \mathbf{\Lambda} \mathbf{u} = \sum_{i=1}^{d} \lambda_i u_i^2, \qquad \text{s.t.} \sum_{i=1}^{d} u_i^2 = 1.$$

Since $\lambda_1 \geq \cdots \geq \lambda_d \geq 0$. the maximum occurs when $u_1 = 1,\ u_2 = u_3 = \cdots u_d = 0$.

Hence, the maximum value is equal to λ_1 occurring at $\mathbf{u}^* = \mathbf{e}_1$ (standard basis vector)

Mapping back to $\mathbf{w}$:

$$\mathbf{w}^\star = \mathbf{Q}\mathbf{u}^\star = \mathbf{Q}\mathbf{e}_1 = \mathbf{q}_1.$$

$$\boxed{\mathbf{w}^\star = \mathbf{q}_1}$$

The first principal component is the top eigenvector of the covariance matrix **S.**
Thus variance of the projected data along the first principle direction is equal to λ_1.

Top-K Principal Components

First principal component: The direction capturing the maximum variance of the data

$$\max_{\|\mathbf{w}\|=1} \mathbf{w}^\top \mathbf{S}\mathbf{w}$$

is achieved at

$$\mathbf{w}^\star = \mathbf{q}_1,$$

the top eigenvector the covariance matrix $\mathbf{S}$

Second principal component: Now restrict to directions orthogonal to $\mathbf{q}_1$:

$$\max_{\|\mathbf{w}\|=1,\, \mathbf{w}^\top \mathbf{q}_1 = 0} \mathbf{w}^\top \mathbf{S}\mathbf{w}.$$

This maximum occurs at $\mathbf{w}^\star = \mathbf{q}_2$, the eigenvector corresponding to λ_2.

Continuing inductively, the k-th principal component is $\mathbf{w}_k = \mathbf{q}_k$, the eigenvector of $\mathbf{S}$ corresponding to the k-th largest eigenvalue λ_k.

Top-K principal components: $\mathbf{Q}_K = [\mathbf{q}_1 \quad \mathbf{q}_2 \quad \cdots \quad \mathbf{q}_K]$.

The total variance captured by the top-K principal components is $\sum_{i=1}^{K} \lambda_i$

PCA Algorithm

Input: Data matrix $\mathbf{X} \in \mathbb{R}^{n \times d}$, target dimension $k < d$

Step 1: Center the data by subtracting the sample mean $\boldsymbol{\mu} = \frac{1}{n}\sum_{i=1}^{n} \mathbf{x}_i$ from each input vector

Step 2: Compute covariance matrix of the centered data $\mathbf{X}$

$$\mathbf{S} = \frac{1}{n}\mathbf{X}^T\mathbf{X}$$

Step 3: Compute Eigen-decomposition of the covariance matrix

$$\mathbf{S} = \mathbf{Q}\boldsymbol{\Lambda}\mathbf{Q}^T$$

Step 4: Select first k eigen vectors corresponding to largest eigen values

$$\mathbf{W} = [\mathbf{q}_1 \quad \mathbf{q}_2 \quad \cdots \quad \mathbf{q}_K]$$

Step 5: Project the data

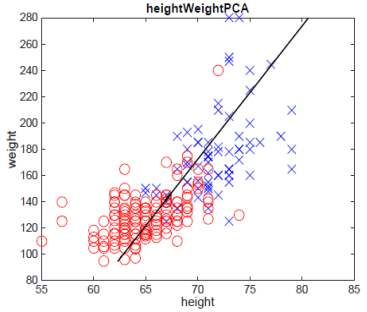
$$\mathbf{Z} = \mathbf{X}\,\mathbf{W}$$

Output: Low-dimensional representation $\mathbf{Z} \in \mathbb{R}^{n \times k}$

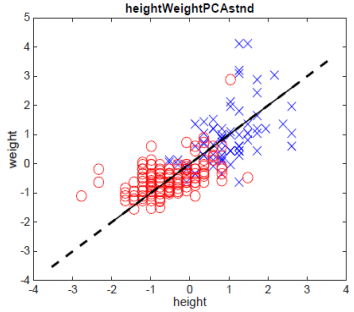
Reconstruction: $\mathbf{X} \approx \widehat{\mathbf{X}} = \mathbf{Z}\,\mathbf{W}^{\mathbf{T}}$

Practical Aspects of PCA

Scale Sensitivity of PCA



(a)



(b)

Figure 20.5: Effect of standardization on PCA applied to the height/weight dataset. (Red=female, blue=male.) Left: PCA of raw data. Right: PCA of standardized data. Generated by pcaStandardization.ipynb.

Raw data (Left):
Because weight varies more than height, the principal direction is biased toward the weight axis.

After standardization (Right):
The principal direction reflects their joint pattern of variation rather than scale differences.

PCA is scale-dependent: variables with larger variance dominate the principal directions.
Recommendation: Standardize when variables have different units or scales.

PCA computation:
Raw data ⇒ Eigen-decomposition of **covariance matrix**
Standardized data ⇒ Eigen-decomposition of **correlation matrix**

Fig Source: PML book by Murphy

Choosing the Number of Components K

Recall that the variance captured by k^{th} principle direction is equal to λ_k

$$\text{Cumulative Explained Variance}(K) = \frac{\sum_{j=1}^{K} \lambda_j}{\sum_{j=1}^{d} \lambda_j}$$

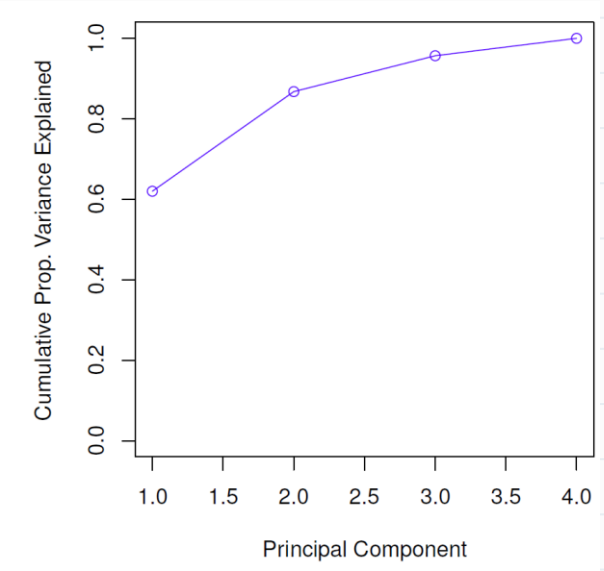
- Total variance captured by first K principal components: $\sum_{j=1}^{K} \lambda_j$
- Total variance of data = $\sum_{j=1}^{d} \lambda_j$

Choose smallest K such that:

$$\frac{\sum_{j=1}^{K} \lambda_j}{\sum_{j=1}^{d} \lambda_j} \geq \tau$$

where $\tau \in (0,1)$ is a user-specified variance retention level..

E.g. $\tau = 0.9$ retains 90% of total variance.



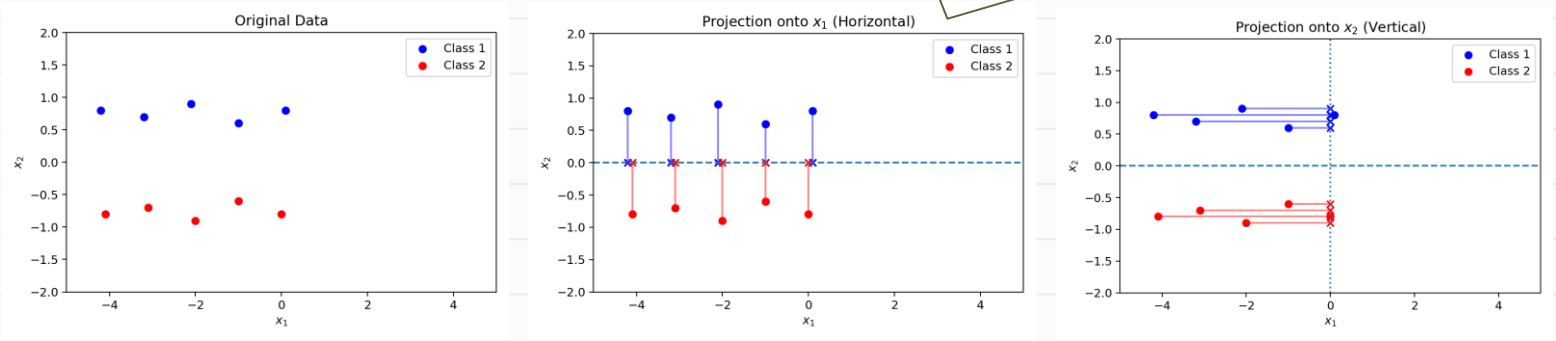
PCA for Feature Extraction in Supervised Learning

PCA can be used as a **feature extraction step** before supervised learning.

Example: High-dimensional images (e.g., 64×64) can be projected onto a low-dimensional PCA subspace and then fed into a classifier (e.g., SVM / logistic regression).

Limitation: PCA maximizes variance, **not class separability**

Largest variance direction is horizontal
Class separation is vertical
PCA discards predictive information



Alternatives (Supervised Dimensionality Reduction):

- **LDA (Linear Discriminant Analysis)**
  → Finds directions that separate different classes as much as possible while keeping same-class points close.

- **PLS (Partial Least Squares)**
  → Finds directions that maximize covariance with the target variable

Computing PCA in Practice

Gram Matrix (when $d \gg n$, e.g., image data**):**

Instead of computing the EVD of the Covariance matrix

$$\mathbf{S} = \frac{1}{n}\mathbf{X}^T\mathbf{X} \in \mathbb{R}^{d\times d},$$

form the gram matrix

$$\mathbf{G} = \frac{1}{n}\mathbf{X}\mathbf{X}^T \in \mathbb{R}^{n\times n}.$$

and compute its eigen vectors $\mathbf{G}\mathbf{u}_i = \lambda_i\mathbf{u}_i$.

Recover principal directions (via left–right singular vector relation):

$$\mathbf{w}_i = \frac{1}{\sqrt{\lambda_i}}\mathbf{X}^T\mathbf{u}_i \in \mathbb{R}^d.$$

SVD (Preferred in Practice)

Compute

$$\mathbf{X} = \mathbf{U}\mathbf{\Sigma}\mathbf{V}^T.$$

Since the eigenvectors of $\mathbf{X}^T\mathbf{X}$ are the right singular vectors of $\mathbf{X}$, we can directly compute SVD of $\mathbf{X}$

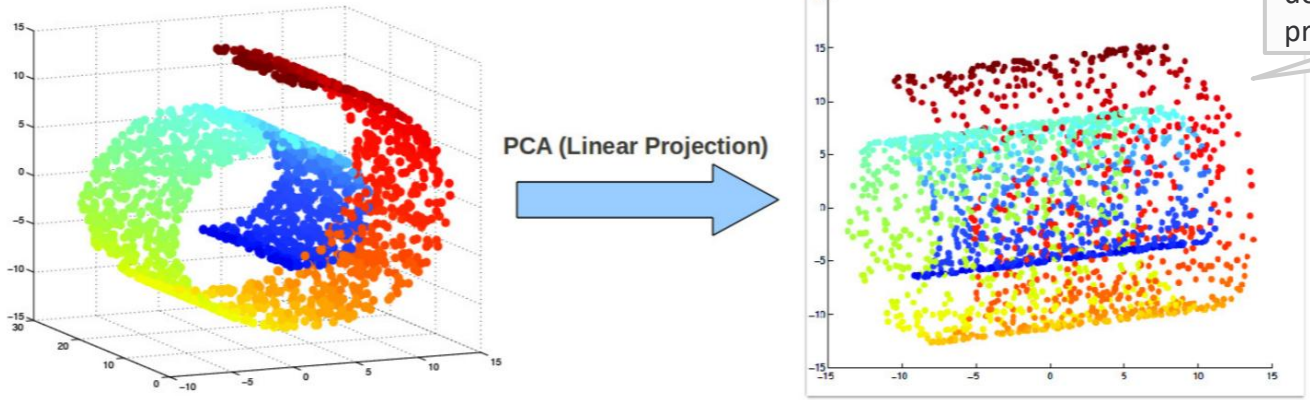
- No explicit covariance or Gram matrix formation
- Numerically more stable

Nonlinear Dimensionality Reduction

Slides adapted from Prof Piyush Rai

Beyond Linear Projections

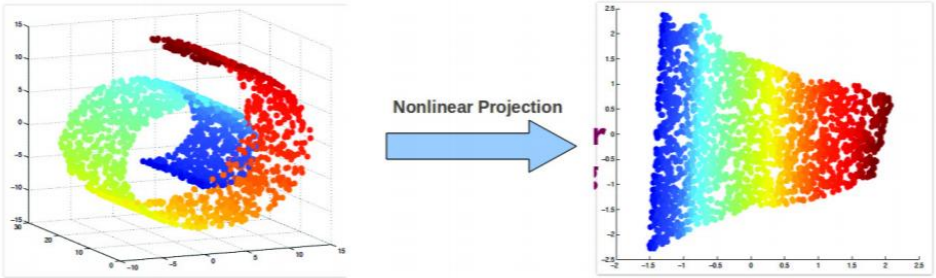
- Consider the swiss-roll dataset (points lying close to a manifold)



- Linear projection methods (e.g., PCA) can't capture intrinsic nonlinearities
  - Maximum variance directions may not be the most interesting ones

Nonlinear Dimensionality Reduction

- We want to a learn nonlinear low-dim projection

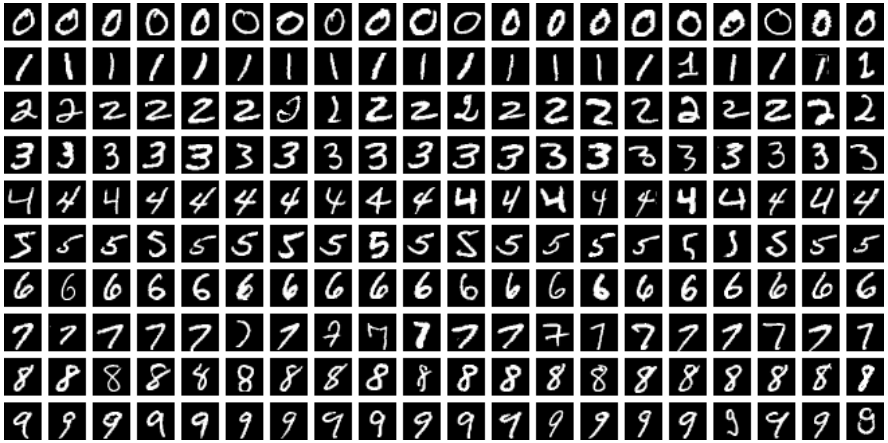


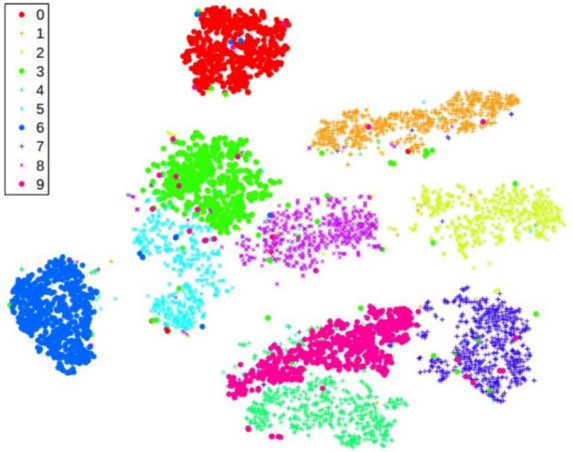
Relative positions of points preserved after the projection

- Some ways of doing this
  - Nonlinearize a linear dimensionality reduction method. E.g.:
    - Cluster data and apply linear PCA within each cluster (mixture of PCA)
    - Kernel PCA (nonlinear PCA)
  - Using manifold based methods that intrinsically preserve nonlinear geometry, e.g.,
    - Locally Linear Embedding (LLE), Isomap
    - SNE, tSNE
- .. or use unsupervised deep learning techniques

Nonlinear Dimensionality Reductions: SNE and t-SNE

Especially useful for visualizing data by projecting into 2D or 3D



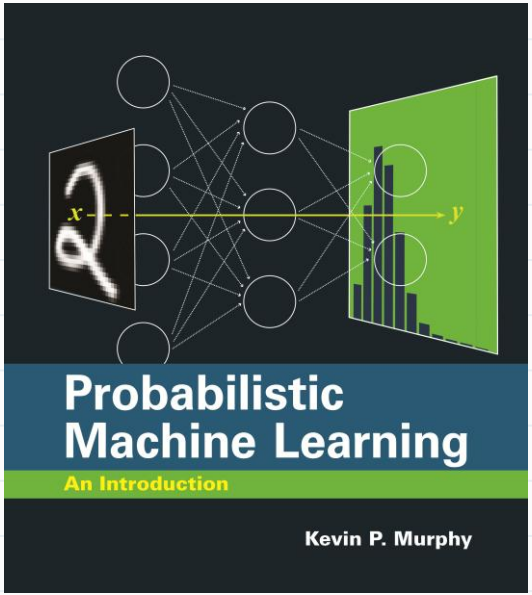


Result of visualizing MNIST digits data (28×28**) in 2D** (Figure from van der Maaten and Hinton, 2008)

Reference

Chapter 20

of “Probabilistic Machine Learning: An Introduction” book by Kevin P. Murphy



Thank You