Principal Component Analysis

CSci 5525: Machine Learning

Instructor: Arindam Banerjee

Dec 3, 2008

The Main Idea

- Given a dataset $\mathcal{X} = \{\mathbf{x}_1, \ldots, \mathbf{x}_N\}$

The Main Idea

- Given a dataset $\mathcal{X} = \{\mathbf{x}_1, \ldots, \mathbf{x}_N\}$
- Find a low-dimensional linear projection

The Main Idea

- Given a dataset $\mathcal{X} = \{\mathbf{x}_1, \ldots, \mathbf{x}_N\}$
- Find a low-dimensional linear projection
- Two possible formulations

The Main Idea

- Given a dataset $\mathcal{X} = \{\mathbf{x}_1, \ldots, \mathbf{x}_N\}$
- Find a low-dimensional linear projection
- Two possible formulations
    - The variance in low-d is maximized

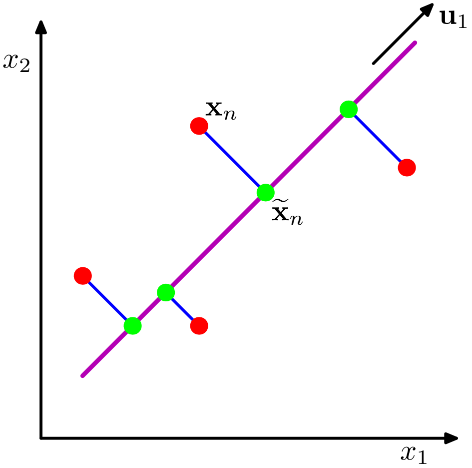
The Main Idea

- Given a dataset $\mathcal{X} = \{\mathbf{x}_1, \ldots, \mathbf{x}_N\}$
- Find a low-dimensional linear projection
- Two possible formulations
  - The variance in low-d is maximized
  - The average projection cost is minimized

The Main Idea

- Given a dataset $\mathcal{X} = \{\mathbf{x}_1, \ldots, \mathbf{x}_N\}$
- Find a low-dimensional linear projection
- Two possible formulations
  - The variance in low-d is maximized
  - The average projection cost is minimized
- Both are equivalent

Two viewpoints



Maximum Variance Formulation

- Consider $\mathcal{X} = \{\mathbf{x}_1, \ldots, \mathbf{x}_N\}$

Maximum Variance Formulation

- Consider $\mathcal{X} = \{\mathbf{x}_1, \ldots, \mathbf{x}_N\}$
- With $\mathbf{x}_i \in \mathbb{R}^d$, goal is to get a projection in $\mathbb{R}^m$, $m < d$

Maximum Variance Formulation

- Consider $\mathcal{X} = \{\mathbf{x}_1, \ldots, \mathbf{x}_N\}$
- With $\mathbf{x}_i \in \mathbb{R}^d$, goal is to get a projection in $\mathbb{R}^m$, $m < d$
- Consider $m = 1$, need a projection vector $\mathbf{u}_1 \in \mathbb{R}^d$

Maximum Variance Formulation

- Consider $\mathcal{X} = \{\mathbf{x}_1, \ldots, \mathbf{x}_N\}$
- With $\mathbf{x}_i \in \mathbb{R}^d$, goal is to get a projection in $\mathbb{R}^m$, $m < d$
- Consider $m = 1$, need a projection vector $\mathbf{u}_1 \in \mathbb{R}^d$
- Each datapoint $\mathbf{x}_i$ gets projected to $\mathbf{u}^T \mathbf{x}_i$

Maximum Variance Formulation

- Consider $\mathcal{X} = \{\mathbf{x}_1, \ldots, \mathbf{x}_N\}$
- With $\mathbf{x}_i \in \mathbb{R}^d$, goal is to get a projection in $\mathbb{R}^m$, $m < d$
- Consider $m = 1$, need a projection vector $\mathbf{u}_1 \in \mathbb{R}^d$
- Each datapoint $\mathbf{x}_i$ gets projected to $\mathbf{u}^T\mathbf{x}_i$
- Mean of the projected data $\mathbf{u}_1^T\bar{\mathbf{x}}$ where

$$\bar{\mathbf{x}} = \frac{1}{N}\sum_{n=1}^{N}\mathbf{x}_n$$

Maximum Variance Formulation

- Consider $\mathcal{X} = \{\mathbf{x}_1, \ldots, \mathbf{x}_N\}$
- With $\mathbf{x}_i \in \mathbb{R}^d$, goal is to get a projection in $\mathbb{R}^m$, $m < d$
- Consider $m = 1$, need a projection vector $\mathbf{u}_1 \in \mathbb{R}^d$
- Each datapoint $\mathbf{x}_i$ gets projected to $\mathbf{u}^T \mathbf{x}_i$
- Mean of the projected data $\mathbf{u}_1^T \bar{\mathbf{x}}$ where

$$\bar{\mathbf{x}} = \frac{1}{N} \sum_{n=1}^{N} \mathbf{x}_n$$

- Variance of the projected data

$$\frac{1}{N} \sum_{n=1}^{N} (\mathbf{u}_1^T \mathbf{x}_n - \mathbf{u}_1^T \bar{\mathbf{x}})^2 = \mathbf{u}_1^T S \mathbf{u}_1$$

where

$$S = \frac{1}{N} \sum_{n=1}^{N} (\mathbf{x}_n - \bar{x})(\mathbf{x}_n - \bar{\mathbf{x}})^T$$

Maximum Variance Formulation (Contd.)

- Maximize $\mathbf{u}_1^T S \mathbf{u}_1$ w.r.t. $\mathbf{u}_1$

Maximum Variance Formulation (Contd.)

- Maximize $\mathbf{u}_1^T S \mathbf{u}_1$ w.r.t. $\mathbf{u}_1$
- Need to have a constraint to prevent $||\mathbf{u}_1|| \rightarrow \infty$

Maximum Variance Formulation (Contd.)

- Maximize $\mathbf{u}_1^T S \mathbf{u}_1$ w.r.t. $\mathbf{u}_1$
- Need to have a constraint to prevent $||\mathbf{u}_1|| \to \infty$
- Normalization constraint $||\mathbf{u}_1||^2 = 1$

Maximum Variance Formulation (Contd.)

- Maximize $\mathbf{u}_1^T S \mathbf{u}_1$ w.r.t. $\mathbf{u}_1$
- Need to have a constraint to prevent $||\mathbf{u}_1|| \to \infty$
- Normalization constraint $||\mathbf{u}_1||^2 = 1$
- The Lagrangian for the problem

$$\mathbf{u}_1^T S \mathbf{u}_1 + \lambda_1 (1 - \mathbf{u}_1^T \mathbf{u}_1)$$

Maximum Variance Formulation (Contd.)

- Maximize $\mathbf{u}_1^T S \mathbf{u}_1$ w.r.t. $\mathbf{u}_1$
- Need to have a constraint to prevent $||\mathbf{u}_1|| \rightarrow \infty$
- Normalization constraint $||\mathbf{u}_1||^2 = 1$
- The Lagrangian for the problem

$$\mathbf{u}_1^T S \mathbf{u}_1 + \lambda_1 (1 - \mathbf{u}_1^T \mathbf{u}_1)$$

- First order necessary condition

$$S \mathbf{u}_1 = \lambda \mathbf{u}_1$$

Maximum Variance Formulation (Contd.)

- Maximize $\mathbf{u}_1^T S \mathbf{u}_1$ w.r.t. $\mathbf{u}_1$
- Need to have a constraint to prevent $||\mathbf{u}_1|| \to \infty$
- Normalization constraint $||\mathbf{u}_1||^2 = 1$
- The Lagrangian for the problem

$$\mathbf{u}_1^T S \mathbf{u}_1 + \lambda_1 (1 - \mathbf{u}_1^T \mathbf{u}_1)$$

- First order necessary condition

$$S \mathbf{u}_1 = \lambda \mathbf{u}_1$$

- $\mathbf{u}_1$ must be 'largest' eigenvector of S since

$$\mathbf{u}_1^T S \mathbf{u}_1 = \lambda_1$$

Maximum Variance Formulation (Contd.)

- Maximize $\mathbf{u}_1^T S \mathbf{u}_1$ w.r.t. $\mathbf{u}_1$
- Need to have a constraint to prevent $||\mathbf{u}_1|| \rightarrow \infty$
- Normalization constraint $||\mathbf{u}_1||^2 = 1$
- The Lagrangian for the problem

$$\mathbf{u}_1^T S \mathbf{u}_1 + \lambda_1 (1 - \mathbf{u}_1^T \mathbf{u}_1)$$

- First order necessary condition

$$S \mathbf{u}_1 = \lambda \mathbf{u}_1$$

- $\mathbf{u}_1$ must be 'largest' eigenvector of S since

$$\mathbf{u}_1^T S \mathbf{u}_1 = \lambda_1$$

- The eigenvector $\mathbf{u}_1$ is called a principal component

Maximum Variance Formulation (Contd.)

- Subsequent principal components must be orthogonal to $\mathbf{u}_1$

Maximum Variance Formulation (Contd.)

- Subsequent principal components must be orthogonal to $\mathbf{u}_1$
- Maximize $\mathbf{u}_2^T S \mathbf{u}_2$ s.t. $||\mathbf{u}_2||^2 = 1, \mathbf{u}_2 \perp \mathbf{u}_1$

Maximum Variance Formulation (Contd.)

- Subsequent principal components must be orthogonal to $\mathbf{u}_1$
- Maximize $\mathbf{u}_2^T S \mathbf{u}_2$ s.t. $||\mathbf{u}_2||^2 = 1, \mathbf{u}_2 \perp \mathbf{u}_1$
- Turns out to be the second eigenvector, and so on

Maximum Variance Formulation (Contd.)

- Subsequent principal components must be orthogonal to $\mathbf{u}_1$
- Maximize $\mathbf{u}_2^T S \mathbf{u}_2$ s.t. $||\mathbf{u}_2||^2 = 1, \mathbf{u}_2 \perp \mathbf{u}_1$
- Turns out to be the second eigenvector, and so on
- The top-m eigenvectors give the 'best' m-dimensional projection

Minimum Error Formulation

- Consider a complete basis $\{\mathbf{u}_i\}$ in $\mathbb{R}^d$

Minimum Error Formulation

- Consider a complete basis $\{\mathbf{u}_i\}$ in $\mathbb{R}^d$
- Each data point can be written as $\mathbf{x}_n = \sum_{i=1}^{d} \alpha_{ni} \mathbf{u}_i$

Minimum Error Formulation

- Consider a complete basis $\{\mathbf{u}_i\}$ in $\mathbb{R}^d$
- Each data point can be written as $\mathbf{x}_n = \sum_{i=1}^{d} \alpha_{ni}\mathbf{u}_i$
- Note that $\alpha_{ni} = \mathbf{x}_n^T \mathbf{u}_i$ so that

$$\mathbf{x}_n = \sum_{i=1}^{d} (\mathbf{x}_n^T \mathbf{u}_i)\mathbf{u}_i$$

Minimum Error Formulation

- Consider a complete basis $\{\mathbf{u}_i\}$ in $\mathbb{R}^d$
- Each data point can be written as $\mathbf{x}_n = \sum_{i=1}^{d} \alpha_{ni} \mathbf{u}_i$
- Note that $\alpha_{ni} = \mathbf{x}_n^T \mathbf{u}_i$ so that

$$\mathbf{x}_n = \sum_{i=1}^{d} (\mathbf{x}_n^T \mathbf{u}_i) \mathbf{u}_i$$

- Our goal is to obtain a lower dimensional subspace $m < d$

Minimum Error Formulation

- Consider a complete basis $\{\mathbf{u}_i\}$ in $\mathbb{R}^d$
- Each data point can be written as $\mathbf{x}_n = \sum_{i=1}^{d} \alpha_{ni}\mathbf{u}_i$
- Note that $\alpha_{ni} = \mathbf{x}_n^T\mathbf{u}_i$ so that

$$\mathbf{x}_n = \sum_{i=1}^{d} (\mathbf{x}_n^T\mathbf{u}_i)\mathbf{u}_i$$

- Our goal is to obtain a lower dimensional subspace $m < d$
- A generic representation of a low-d point

$$\tilde{\mathbf{x}}_n = \sum_{i=1}^{m} z_{ni}\mathbf{u}_i + \sum_{i=m+1}^{d} b_i\mathbf{u}_i$$

Minimum Error Formulation

- Consider a complete basis $\{\mathbf{u}_i\}$ in $\mathbb{R}^d$
- Each data point can be written as $\mathbf{x}_n = \sum_{i=1}^{d} \alpha_{ni}\mathbf{u}_i$
- Note that $\alpha_{ni} = \mathbf{x}_n^T\mathbf{u}_i$ so that

$$\mathbf{x}_n = \sum_{i=1}^{d} (\mathbf{x}_n^T\mathbf{u}_i)\mathbf{u}_i$$

- Our goal is to obtain a lower dimensional subspace $m < d$
- A generic representation of a low-d point

$$\tilde{\mathbf{x}}_n = \sum_{i=1}^{m} z_{ni}\mathbf{u}_i + \sum_{i=m+1}^{d} b_i\mathbf{u}_i$$

- Coefficients z_{ni} depend on the data point $\mathbf{x}_n$

Minimum Error Formulation

- Consider a complete basis $\{\mathbf{u}_i\}$ in $\mathbb{R}^d$
- Each data point can be written as $\mathbf{x}_n = \sum_{i=1}^{d} \alpha_{ni} \mathbf{u}_i$
- Note that $\alpha_{ni} = \mathbf{x}_n^T \mathbf{u}_i$ so that

$$\mathbf{x}_n = \sum_{i=1}^{d} (\mathbf{x}_n^T \mathbf{u}_i) \mathbf{u}_i$$

- Our goal is to obtain a lower dimensional subspace $m < d$
- A generic representation of a low-d point

$$\tilde{\mathbf{x}}_n = \sum_{i=1}^{m} z_{ni} \mathbf{u}_i + \sum_{i=m+1}^{d} b_i \mathbf{u}_i$$

- Coefficients z_{ni} depend on the data point $\mathbf{x}_n$
- Free to choose $z_{ni}, b_i, \mathbf{u}_i$ to get $\tilde{\mathbf{x}}_n$ close to $\mathbf{x}_n$

Minimum Error Formulation (Contd.)

- The objective is to minimize

$$J = \frac{1}{N} \sum_{n=1}^{N} \|\mathbf{x}_n - \tilde{\mathbf{x}}_n\|^2$$

Minimum Error Formulation (Contd.)

- The objective is to minimize

$$J = \frac{1}{N}\sum_{n=1}^{N} \|\mathbf{x}_n - \tilde{\mathbf{x}}_n\|^2$$

- Taking derivative w.r.t. z_{ni} we get $z_{nj} = \mathbf{x}_n^T \mathbf{u}_j, j = 1, \ldots, m$

Minimum Error Formulation (Contd.)

- The objective is to minimize

$$J = \frac{1}{N} \sum_{n=1}^{N} \|\mathbf{x}_n - \tilde{\mathbf{x}}_n\|^2$$

- Taking derivative w.r.t. z_{ni} we get $z_{nj} = \mathbf{x}_n^T \mathbf{u}_j, j = 1, \ldots, m$
- Taking derivative w.r.t. b_j we get $b_j = \bar{\mathbf{x}}^T \mathbf{u}_j, j = m+1, \ldots, d$

Minimum Error Formulation (Contd.)

- The objective is to minimize

$$J = \frac{1}{N} \sum_{n=1}^{N} \|\mathbf{x}_n - \tilde{\mathbf{x}}_n\|^2$$

- Taking derivative w.r.t. z_{ni} we get $z_{nj} = \mathbf{x}_n^T \mathbf{u}_j, j = 1, \ldots, m$
- Taking derivative w.r.t. b_j we get $b_j = \bar{\mathbf{x}}^T \mathbf{u}_j, j = m+1, \ldots, d$
- Then we have

$$\mathbf{x}_n - \tilde{\mathbf{x}}_n = \sum_{i=m+1}^{d} \{(\mathbf{x}_n - \bar{\mathbf{x}})^T \mathbf{u}_i\} \mathbf{u}_i$$

Minimum Error Formulation (Contd.)

- The objective is to minimize

$$J = \frac{1}{N} \sum_{n=1}^{N} \|\mathbf{x}_n - \tilde{\mathbf{x}}_n\|^2$$

- Taking derivative w.r.t. z_{ni} we get $z_{nj} = \mathbf{x}_n^T \mathbf{u}_j, j = 1, \ldots, m$
- Taking derivative w.r.t. b_j we get $b_j = \bar{\mathbf{x}}^T \mathbf{u}_j, j = m+1, \ldots, d$
- Then we have

$$\mathbf{x}_n - \tilde{\mathbf{x}}_n = \sum_{i=m+1}^{d} \{(\mathbf{x}_n - \bar{\mathbf{x}})^T \mathbf{u}_i\} \mathbf{u}_i$$

- Lies in the space orthogonal to the principal subspace

Minimum Error Formulation (Contd.)

- The objective is to minimize

$$J = \frac{1}{N}\sum_{n=1}^{N} \|\mathbf{x}_n - \tilde{\mathbf{x}}_n\|^2$$

- Taking derivative w.r.t. z_{ni} we get $z_{nj} = \mathbf{x}_n^T \mathbf{u}_j, j = 1, \ldots, m$
- Taking derivative w.r.t. b_j we get $b_j = \bar{\mathbf{x}}^T \mathbf{u}_j, j = m+1, \ldots, d$
- Then we have

$$\mathbf{x}_n - \tilde{\mathbf{x}}_n = \sum_{i=m+1}^{d} \{(\mathbf{x}_n - \bar{\mathbf{x}})^T \mathbf{u}_i\}\mathbf{u}_i$$

- Lies in the space orthogonal to the principal subspace
- The distortion measure to be minimized

$$J = \frac{1}{N}\sum_{n=1}^{N}\sum_{i=m+1}^{d} (\mathbf{x}_n^T\mathbf{u}_i - \bar{\mathbf{x}}^T\mathbf{u}_i)^2 = \sum_{i=m+1}^{d} \mathbf{u}_i^T S \mathbf{u}_i$$

Minimum Error Formulation (Contd.)

- The objective is to minimize

$$J = \frac{1}{N}\sum_{n=1}^{N} ||\mathbf{x}_n - \tilde{\mathbf{x}}_n\|^2$$

- Taking derivative w.r.t. z_{ni} we get $z_{nj} = \mathbf{x}_n^T \mathbf{u}_j, j = 1, \ldots, m$
- Taking derivative w.r.t. b_j we get $b_j = \bar{\mathbf{x}}^T \mathbf{u}_j, j = m+1, \ldots, d$
- Then we have

$$\mathbf{x}_n - \tilde{\mathbf{x}}_n = \sum_{i=m+1}^{d} \{(\mathbf{x}_n - \bar{\mathbf{x}})^T \mathbf{u}_i\}\mathbf{u}_i$$

- Lies in the space orthogonal to the principal subspace
- The distortion measure to be minimized

$$J = \frac{1}{N}\sum_{n=1}^{N}\sum_{i=m+1}^{d} (\mathbf{x}_n^T \mathbf{u}_i - \bar{\mathbf{x}}^T \mathbf{u}_i)^2 = \sum_{i=m+1}^{d} \mathbf{u}_i^T S \mathbf{u}_i$$

- Need orthonormality constraints on $\mathbf{u}_i$ to prevent $\mathbf{u}_i = 0$ solution

Minimum Error Formulation (Contd.)

- Consider special case $d = 2, m = 1$

Minimum Error Formulation (Contd.)

- Consider special case $d = 2, m = 1$
- The Lagrangian of the objective

$$L = \mathbf{u}_2^T S \mathbf{u}_2 + \lambda_2 (1 - \mathbf{u}_2^T \mathbf{u}_2)$$

Minimum Error Formulation (Contd.)

- Consider special case $d = 2, m = 1$
- The Lagrangian of the objective

$$L = \mathbf{u}_2^T S \mathbf{u}_2 + \lambda_2 (1 - \mathbf{u}_2^T \mathbf{u}_2)$$

- First order condition is $S\mathbf{u}_2 = \lambda_2 \mathbf{u}_2$

Minimum Error Formulation (Contd.)

- Consider special case $d = 2, m = 1$
- The Lagrangian of the objective

$$L = \mathbf{u}_2^T S \mathbf{u}_2 + \lambda_2 (1 - \mathbf{u}_2^T \mathbf{u}_2)$$

- First order condition is $S\mathbf{u}_2 = \lambda_2 \mathbf{u}_2$
- In general, the condition is $S\mathbf{u}_i = \lambda_i \mathbf{u}_i$

Minimum Error Formulation (Contd.)

- Consider special case $d = 2, m = 1$
- The Lagrangian of the objective

$$L = \mathbf{u}_2^T S \mathbf{u}_2 + \lambda_2 (1 - \mathbf{u}_2^T \mathbf{u}_2)$$

- First order condition is $S\mathbf{u}_2 = \lambda_2 \mathbf{u}_2$
- In general, the condition is $S\mathbf{u}_i = \lambda_i \mathbf{u}_i$
- Given by the eigenvectors corresponding to the smallest $(d - m)$ eigenvalues

Minimum Error Formulation (Contd.)

- Consider special case $d = 2, m = 1$
- The Lagrangian of the objective

$$L = \mathbf{u}_2^T S \mathbf{u}_2 + \lambda_2 (1 - \mathbf{u}_2^T \mathbf{u}_2)$$

- First order condition is $S\mathbf{u}_2 = \lambda_2 \mathbf{u}_2$
- In general, the condition is $S\mathbf{u}_i = \lambda_i \mathbf{u}_i$
- Given by the eigenvectors corresponding to the smallest $(d - m)$ eigenvalues
- So the principal space $\mathbf{u}_i, i = 1, \ldots, m$ are the 'largest' eigenvectors

Kernel PCA

- In PCA, the principal components $\mathbf{u}_i$ are given by

$$S\mathbf{u}_i = \lambda_i \mathbf{u}_i$$

where

$$S = \frac{1}{N}\sum_{n=1}^{N} \mathbf{x}_n \mathbf{x}_n^T$$

- Consider a feature mapping $\phi(\mathbf{x})$
- Want to implicitly perform PCA in the feature space
- Assume the features have zero mean $\sum_n \phi(\mathbf{x}_n) = 0$

Kernel PCA (Contd.)

- The sample covariance matrix in the feature space

$$C = \frac{1}{N} \sum_{n=1}^{N} \phi(\mathbf{x}_n)\phi(\mathbf{x}_n)^T$$

Kernel PCA (Contd.)

- The sample covariance matrix in the feature space

$$C = \frac{1}{N}\sum_{n=1}^{N} \phi(\mathbf{x}_n)\phi(\mathbf{x}_n)^T$$

- The eigenvectors are given by

$$C\mathbf{v}_i = \lambda_i \mathbf{v}_i$$

Kernel PCA (Contd.)

- The sample covariance matrix in the feature space

$$C = \frac{1}{N} \sum_{n=1}^{N} \phi(\mathbf{x}_n)\phi(\mathbf{x}_n)^T$$

- The eigenvectors are given by

$$C\mathbf{v}_i = \lambda_i \mathbf{v}_i$$

- We want to avoid computing C explicitly

Kernel PCA (Contd.)

- The sample covariance matrix in the feature space

$$C = \frac{1}{N}\sum_{n=1}^{N}\phi(\mathbf{x}_n)\phi(\mathbf{x}_n)^T$$

- The eigenvectors are given by

$$C\mathbf{v}_i = \lambda_i \mathbf{v}_i$$

- We want to avoid computing C explicitly
- Note that the eigenvectors satisfy

$$\frac{1}{N}\sum_{n=1}^{N}\phi(\mathbf{x}_n)\left\{\phi(\mathbf{x}_n)^T\mathbf{v}_i\right\} = \lambda_i \mathbf{v}_i$$

Kernel PCA (Contd.)

- The sample covariance matrix in the feature space

$$C = \frac{1}{N} \sum_{n=1}^{N} \phi(\mathbf{x}_n) \phi(\mathbf{x}_n)^T$$

- The eigenvectors are given by

$$C \mathbf{v}_i = \lambda_i \mathbf{v}_i$$

- We want to avoid computing C explicitly
- Note that the eigenvectors satisfy

$$\frac{1}{N} \sum_{n=1}^{N} \phi(\mathbf{x}_n) \left\{ \phi(\mathbf{x}_n)^T \mathbf{v}_i \right\} = \lambda_i \mathbf{v}_i$$

- Since the inner product is a scaler, we have

$$v_i = \sum_{n=1}^{N} a_{in} \phi(\mathbf{x}_n)$$

Kernel PCA (Contd.)

- Substituting back into the eigenvalue equation

$$\frac{1}{N}\sum_{n=1}^{N}\phi(\mathbf{x}_n)\phi(\mathbf{x}_n)^T\sum_{m=1}^{N}a_{im}\phi(\mathbf{x}_m) = \lambda_i\sum_{n=1}^{N}a_{in}\phi(\mathbf{x}_n)$$

Kernel PCA (Contd.)

- Substituting back into the eigenvalue equation

$$\frac{1}{N}\sum_{n=1}^{N}\phi(\mathbf{x}_n)\phi(\mathbf{x}_n)^T\sum_{m=1}^{N}a_{im}\phi(\mathbf{x}_m) = \lambda_i\sum_{n=1}^{N}a_{in}\phi(\mathbf{x}_n)$$

- Multiplying both sides by $\phi(\mathbf{x}_l)$ and using $K(\mathbf{x}_n, \mathbf{x}_m) = \phi(\mathbf{x}_n)^T\phi(\mathbf{x}_m)$, we have

$$\frac{1}{N}\sum_{n=1}^{N}K(\mathbf{x}_l, \mathbf{x}_n)\sum_{m=1}^{N}a_{im}K(\mathbf{x}_n, \mathbf{x}_m) = \lambda_i\sum_{n=1}^{N}a_{in}K(\mathbf{x}_l, \mathbf{x}_n)$$

Kernel PCA (Contd.)

- Substituting back into the eigenvalue equation

$$\frac{1}{N}\sum_{n=1}^{N}\phi(\mathbf{x}_n)\phi(\mathbf{x}_n)^T\sum_{m=1}^{N}a_{im}\phi(\mathbf{x}_m)=\lambda_i\sum_{n=1}^{N}a_{in}\phi(\mathbf{x}_n)$$

- Multiplying both sides by $\phi(\mathbf{x}_l)$ and using $K(\mathbf{x}_n,\mathbf{x}_m)=\phi(\mathbf{x}_n)^T\phi(\mathbf{x}_m)$, we have

$$\frac{1}{N}\sum_{n=1}^{N}K(\mathbf{x}_l,\mathbf{x}_n)\sum_{m=1}^{N}a_{im}K(\mathbf{x}_n,\mathbf{x}_m)=\lambda_i\sum_{n=1}^{N}a_{in}K(\mathbf{x}_l,\mathbf{x}_n)$$

- In matrix notation, we have

$$K^2\mathbf{a}_i=\lambda_i NK\mathbf{a}_i$$

Kernel PCA (Contd.)

- Substituting back into the eigenvalue equation

$$\frac{1}{N}\sum_{n=1}^{N}\phi(\mathbf{x}_n)\phi(\mathbf{x}_n)^T\sum_{m=1}^{N}a_{im}\phi(\mathbf{x}_m) = \lambda_i\sum_{n=1}^{N}a_{in}\phi(\mathbf{x}_n)$$

- Multiplying both sides by $\phi(\mathbf{x}_l)$ and using $K(\mathbf{x}_n, \mathbf{x}_m) = \phi(\mathbf{x}_n)^T\phi(\mathbf{x}_m)$, we have

$$\frac{1}{N}\sum_{n=1}^{N}K(\mathbf{x}_l, \mathbf{x}_n)\sum_{m=1}^{N}a_{im}K(\mathbf{x}_n, \mathbf{x}_m) = \lambda_i\sum_{n=1}^{N}a_{in}K(\mathbf{x}_l, \mathbf{x}_n)$$

- In matrix notation, we have

$$K^2\mathbf{a}_i = \lambda_i N K\mathbf{a}_i$$

- Except for eigenvectors with 0 eigenvalues, we can solve

$$K\mathbf{a}_i = \lambda_i N\mathbf{a}_i$$

Kernel PCA (Contd.)

- Since the original $\mathbf{v}_i$ are normalized, we have

$$1 = \mathbf{v}_i^T \mathbf{v}_i = \mathbf{a}_i^T K \mathbf{a}_i = \lambda_i N \mathbf{a}_i^T \mathbf{a}_i$$

- Gives a normalization condition for $\mathbf{a}_i$
- Compute $\mathbf{a}_i$ by solving the eigenvalue decomposition
- The ‘projection’ of a point is given by

$$y_i(\mathbf{x}) = \phi(\mathbf{x}_i)^T \mathbf{v}_i = \sum_{n=1}^{N} a_{in} \phi(\mathbf{x}_n)^T \phi(\mathbf{x}_n) = \sum_{n=1}^{N} a_{in} K(\mathbf{x}, \mathbf{x}_n)$$

Illustration of Kernel PCA (Feature Space)

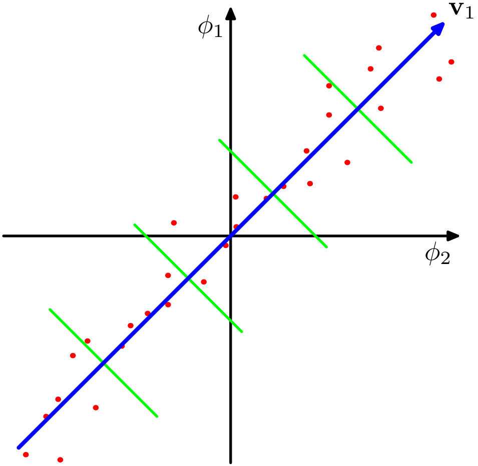
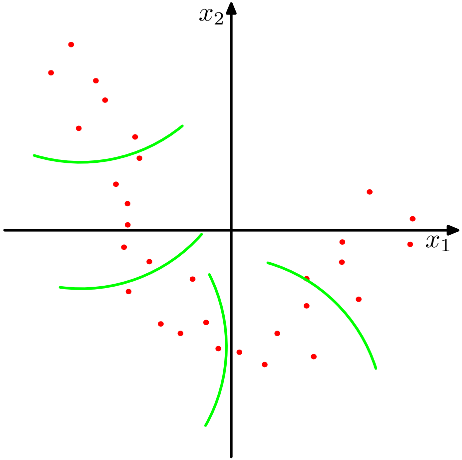


Illustration of Kernel PCA (Data Space)



Dimensionality of Projection

- Original $\mathbf{x}_i \in \mathbb{R}^d$, feature $\phi(\mathbf{x}_i) \in \mathbb{R}^D$
- Possibly $D >> d$ so that the number of principal components can be greater than d
- However, the number of nonzero eigenvalues cannot exceed N
- The covariance matrix C has rank at most N, even if $D >> d$
- Kernel PCA involves eigenvalue decomposition of a $N \times N$ matrix

Kernel PCA: Non-zero Mean

- The features need not have zero mean
- Note that the features cannot be explicitly centered
- The centralized data would be of the form

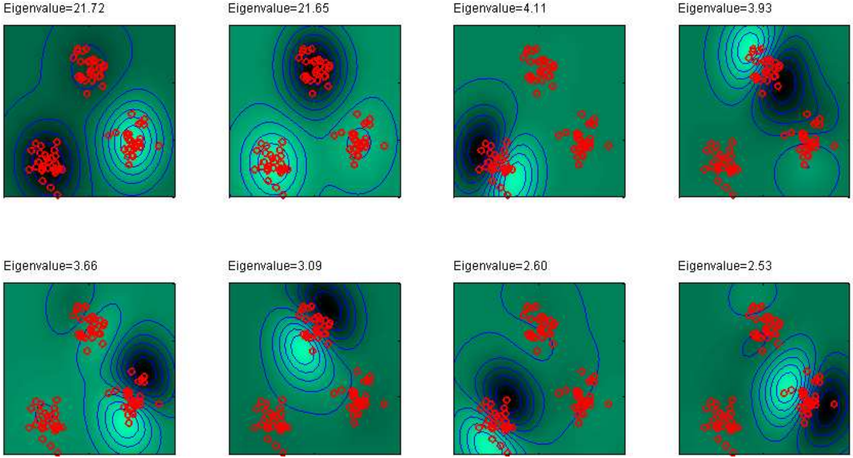
$$\tilde{\phi}(\mathbf{x}_n) = \phi(\mathbf{x}_n) - \frac{1}{N}\sum_{l=1}^{N}\phi(\mathbf{x}_l)$$

- The corresponding gram matrix

$$\tilde{K} = K - 1_N K - K 1_N + 1_N K 1_N$$

- Use $\tilde{K}$ in the basic kernel PCA formulation

Kernel PCA on Artificial Data



Kernel PCA Properties

- Computes eigenvalue decomposition of $N \times N$ matrix
  - Standard PCA computes it for $d \times d$
  - For large datasets $N >> d$, Kernel PCA is more expensive
- Standard PCA gives projection to a low dimensional principal subspace

$$\hat{\mathbf{x}}_n = \sum_{i=1}^{\ell} (\mathbf{x}_n^T \mathbf{u}_i) \mathbf{u}_i$$

- Kernel PCA cannot do this
  - $\phi(\mathbf{x})$ forms a d-dimensional manifold in $\mathbb{R}^D$
  - PCA projection $\hat{\phi}$ of $\phi(\mathbf{x})$ need not be in the manifold
  - May not have a pre-image $\hat{\mathbf{x}}$ in the data space