ECE 595: Machine Learning I
Lecture 7.2: Kernel-PCA

Spring 2020

Stanley Chan

School of Electrical and Computer Engineering
Purdue University

Outline

Feature Analysis

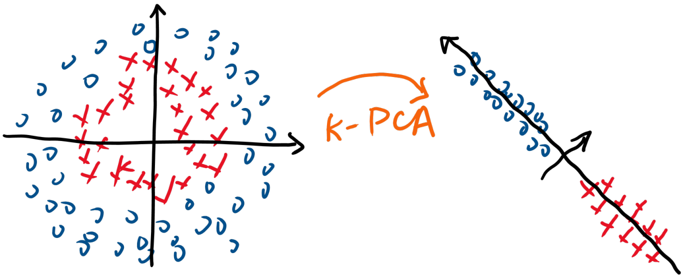
- Lecture 7 Principal Component Analysis (PCA)
- Lecture 8 Hand-Crafted and Deep Features

This Lecture

- PCA
  - Low-dimensional Representation
  - Geometric Interpretation
  - Eigen-Face Problem
- Kernel-PCA
  - Adding kernels to PCA
  - Algorithm
  - Examples

Motivation of Kernel PCA

- Data is originally difficult for PCA
- Find a nonlinear transform
- Idea: Leverage the kernel trick: $k(\boldsymbol{x}^{(i)}, \boldsymbol{x}^{(j)}) = \langle \phi(\boldsymbol{x}^{(i)}), \phi(\boldsymbol{x}^{(j)}) \rangle$
- Example: Left is hard for PCA. After K-PCA, right has a clear principal component.



Kernel for Covariance Matrix

- Assume $\phi(\boldsymbol{x}^{(n)})$ has zero mean. Then consdier the covariance matrix

$$\boldsymbol{\Sigma} = \frac{1}{N}\sum_{n=1}^{N} \boldsymbol{x}^{(n)}(\boldsymbol{x}^{(n)})^T.$$

- Replacing the outer products by feature transforms

$$\boldsymbol{x}^{(n)} \quad \rightarrow \phi(\boldsymbol{x}^{(n)}),$$

  for some nonlinear transformation ϕ.
- If this can be done, then the covariance will become

$$\boldsymbol{\Sigma} = \frac{1}{N}\sum_{n=1}^{N} \phi(\boldsymbol{x}^{(n)})\phi(\boldsymbol{x}^{(n)})^T.$$

- But this is not enough because a kernel needs an inner product

$$k(\boldsymbol{x}^{(n)}, \boldsymbol{x}^{(m)}) = \phi(\boldsymbol{x}^{(n)})^T\phi(\boldsymbol{x}^{(m)}).$$

Kernel Trick

- Recall: PCA solves the eigen-decomposition problem:

$$\boldsymbol{\Sigma}\boldsymbol{u} = \lambda\boldsymbol{u}$$

  So we also need to consider $\boldsymbol{u}$.

- How about this candidate? (Recall: In Kernel Method we express the model parameter as a linear combination of the samples):

$$\boldsymbol{u} = \sum_{n=1}^{N} \alpha_n \phi(\boldsymbol{x}^{(n)}).$$

- Substitute this into the equation $\boldsymbol{\Sigma}\boldsymbol{u} = \lambda\boldsymbol{u}$:

$$\underbrace{\left(\frac{1}{N}\sum_{n=1}^{N} \phi(\boldsymbol{x}^{(n)})\phi(\boldsymbol{x}^{(n)})^T\right)}_{\boldsymbol{\Sigma}} \underbrace{\left(\sum_{m=1}^{N} \alpha_m \phi(\boldsymbol{x}^{(m)})\right)}_{\boldsymbol{u}} = \underbrace{\lambda\left(\sum_{n=1}^{N} \alpha_n \phi(\boldsymbol{x}^{(n)})\right)}_{\lambda\boldsymbol{u}}$$

Kernel Trick

- This means

$$\frac{1}{N}\sum_{n=1}^{N}\phi(\boldsymbol{x}^{(n)})\left(\sum_{m=1}^{N}\alpha_m\phi(\boldsymbol{x}^{(n)})^T\phi(\boldsymbol{x}^{(m)})\right)=\lambda\sum_{n=1}^{N}\alpha_n\phi(\boldsymbol{x}^{(n)})$$

- Recognizing $\phi(\boldsymbol{x}^{(n)})^T\phi(\boldsymbol{x}^{(m)})=k(\boldsymbol{x}^{(n)},\boldsymbol{x}^{(m)})$:

$$\frac{1}{N}\sum_{n=1}^{N}\phi(\boldsymbol{x}^{(n)})\left(\sum_{m=1}^{N}\alpha_n k(\boldsymbol{x}^{(n)},\boldsymbol{x}^{(m)})\right)=\lambda\sum_{n=1}^{N}\alpha_n\phi(\boldsymbol{x}^{(n)})$$

- Multiply $\phi(\boldsymbol{x}^{(\ell)})^T$ on both sides.

$$\frac{1}{N}\sum_{n=1}^{N}k(\boldsymbol{x}^{(\ell)},\boldsymbol{x}^{(n)})\left(\sum_{m=1}^{N}\alpha_n k(\boldsymbol{x}^{(n)},\boldsymbol{x}^{(m)})\right)=\lambda\sum_{n=1}^{N}\alpha_n k(\boldsymbol{x}^{(\ell)},\boldsymbol{x}^{(n)})$$

- This is $\frac{1}{N}\boldsymbol{K}(\boldsymbol{K}\boldsymbol{\alpha})=\lambda\boldsymbol{K}\boldsymbol{\alpha}$.

Eigenvectors of K-PCA

- Rearrange the terms we have that $\boldsymbol{K}^2\boldsymbol{\alpha} = N\lambda\boldsymbol{K}\boldsymbol{\alpha}$.
- We can remove one of the $\boldsymbol{K}$'s since it only causes issues with zero-eigenvalues which are not important to us anyway. So we have

$$\boldsymbol{K}\boldsymbol{\alpha} = N\lambda\boldsymbol{\alpha}. \tag{2}$$

- This is just another eigen-decomposition problem. We moved from $\boldsymbol{\Sigma}\boldsymbol{u} = \lambda\boldsymbol{u}$ to $\boldsymbol{K}\boldsymbol{\alpha} = N\lambda\boldsymbol{\alpha}$. Note that $\boldsymbol{\alpha}$ is the coefficients for $\boldsymbol{u}$:

$$\boldsymbol{u} = \sum_{n=1}^{N} \alpha_n \phi(\boldsymbol{x}^{(n)}) = \boldsymbol{\Phi}\boldsymbol{\alpha},$$

where $\boldsymbol{\Phi} = [\phi(\boldsymbol{x}^{(1)}), \ldots, \phi(\boldsymbol{x}^{(N)})]$ is the transformed data matrix. Recall $\boldsymbol{\Phi}\boldsymbol{\Phi}^T = \boldsymbol{K}$ is the kernel matrix where

$$[\boldsymbol{K}]_{ij} = \phi(\boldsymbol{x}^{(i)})^T\phi(\boldsymbol{x}^{(j)}).$$

Representation in Kernel Space

- If you run eigen-decomposition on $\boldsymbol{K}$, you will get p eigen-vectors $\boldsymbol{\alpha}_1, \ldots, \boldsymbol{\alpha}_p$ where p is the number you choose.
- When a new sample $\boldsymbol{x}$ comes, the j-th representation coefficient is

$$\beta_j = \phi(\boldsymbol{x})^T \boldsymbol{u} = \phi(\boldsymbol{x})^T \sum_{n=1}^{N} \alpha_{jn} \phi(\boldsymbol{x}^{(n)}) = \sum_{n=1}^{N} \alpha_{jn} k(\boldsymbol{x}, \boldsymbol{x}^{(n)}). \quad (3)$$

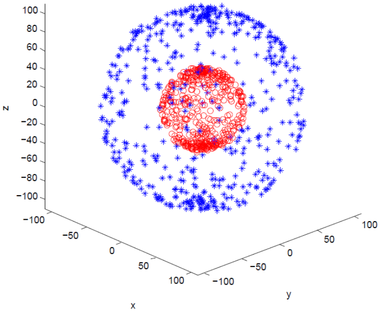
- For the entire representation $\boldsymbol{\beta} \in \mathbb{R}^p$, we have

$$\boldsymbol{\beta} = \begin{bmatrix} ---\boldsymbol{\alpha}_1^T --- \\ \vdots \\ ---\boldsymbol{\alpha}_p^T --- \end{bmatrix} \begin{bmatrix} k(\boldsymbol{x}, \boldsymbol{x}^{(1)}) \\ k(\boldsymbol{x}, \boldsymbol{x}^{(2)}) \\ \vdots \\ k(\boldsymbol{x}, \boldsymbol{x}^{(N)}) \end{bmatrix} \quad (4)$$

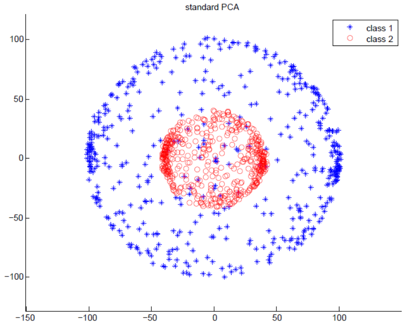
where $\boldsymbol{\alpha}_j = [\alpha_{j1}, \ldots, \alpha_{iN}]^T$.

Example

Here is an example taken from Wang (2012) Kernel Principal Component Analysis and its Applications https://arxiv.org/abs/1207.3538



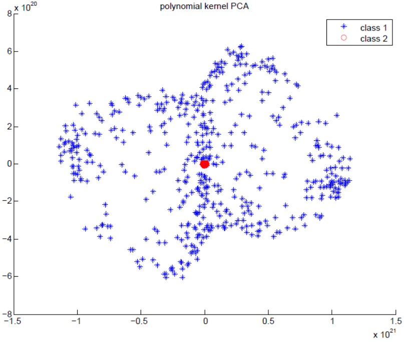
Original Data



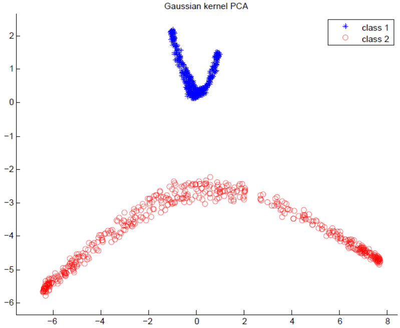
Linear PCA

Example

Here is an example taken from Wang (2012) Kernel Principal Component Analysis and its Applications https://arxiv.org/abs/1207.3538



K-PCA with polynomial



K-PCA with Gaussian

Reading List

PCA Tutorial

- Jonathon Shlens "A Tutorial on Principal Component Analysis", https://arxiv.org/pdf/1404.1100.pdf

PCA: Should We Remove Mean?

- Paul Honeine, "An eigenanalysis of data centering in machine learning", https://arxiv.org/pdf/1407.2904.pdf
- Does mean centering or feature scaling affect a Principal Component Analysis? https://sebastianraschka.com/faq/docs/pca-scaling.html

K-PCA

- Quan Wang (2012), "Kernel Principal Component Analysis and its Applications", https://arxiv.org/abs/1207.3538
- Schölkopf et al. (2005), "Kernel Principal Component Analysis", https://link.springer.com/chapter/10.1007/BFb0020217