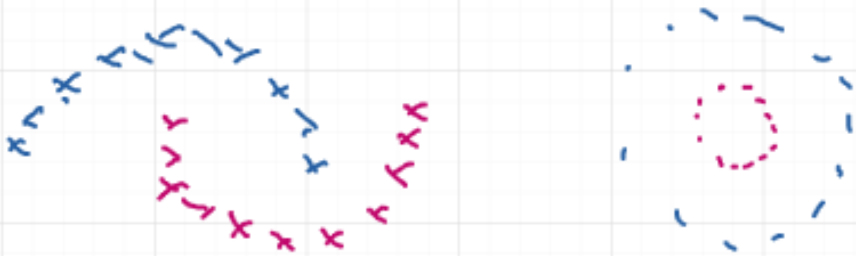
Kernel PCA

Two moons dataset



Note

1) We're assume mean-centered data

$$EX = 0$$

2) $\Sigma_n = \frac{1}{n} \sum_{i=1}^{n} x_i x_i^T$ (biased)

Consider a 2D dataset $x = (x_1, x_2)$

We can lift the data to 6d:

$$\phi(x) = [1, x_1, x_2, x_1^2, x_2^2, x_1 x_2] \in \mathbb{R}^6$$

Instead of applying PCA to X

We apply PCA to $\phi(X)$

$$R_\phi = \frac{1}{n} \sum_i \phi(x_i) \phi(x_i)^T \in \mathbb{R}^{l \times l}$$

Motivation: In the high dim space we can find linear subspace (principal directions) that capture the variability of our data.

Reminder:

If u is a vector in the span of $\{X\}$ we can write u as a linear combination of the vectors x_i in X

Kernel Trick

Let v be an eigenvector of R_ϕ

We can write

$$v \in \text{span}\, \phi(X) \Rightarrow v = \sum_{j=1}^{n} \alpha_j \phi(x_j)$$

$$R_\phi v = \lambda v$$

$$\underbrace{\frac{1}{n} \sum_{i} \phi(x_i)\phi(x_i)^T}_{R_\phi} \sum_{j=1}^{n} \alpha_j \phi(x_j) = \lambda \sum_{i=1}^{n} \alpha_i \phi(x_i)$$

Multiply $\phi(x_k)^T$ on the left and rearrange sums:

$$\frac{1}{n}\sum_{j=1}^{n}\alpha_j\sum_{i=1}^{n}\phi(x_k)^T\phi(x_i)\,\phi(x_i)^T\phi(x_j) = \lambda\sum_{i=1}^{n}\alpha_i\,\phi(x_k)^T\phi(x_i)$$

$$K_{ij} = \phi(x_i)^T\phi(x_j) \quad \text{(kernel)}$$

$$\frac{1}{n}\sum_{j=1}^{n}\alpha_j\sum_{i=1}^{n}K_{ij}K_{ki} = \lambda\sum_{i=1}^{n}\alpha_i K_{ki}$$

$$\Rightarrow \frac{1}{n}KK\alpha = \frac{1}{n}\boxed{K^2\alpha = \lambda K\alpha}$$

$$K(K\alpha - \lambda n\alpha) = 0$$

$\Rightarrow$ α is an EV of K

Projecting onto EV v of R_ϕ (PCA)

$$\phi(x)^T v = \sum_{j=1}^{n}\alpha_j\,\phi(x)^T\phi(x_j) = \sum_{j=1}^{n}K(x,x_j)\alpha_j$$

$$= \underbrace{K\alpha}_{K = k(x,x_j).}\qquad x_j \in X$$

Kernel PCA elevates from p to $l \gg p$ and then reduce dim to d

Instead we calculate kernel matrix $(n \times n)$ and its eigen decomposition gives the "non linear" principal components

---

Kernel matrix = Gram matrix

The Gram matrix of X is given by

$$G = X^T X, \quad G_{ij} = \langle x_i, x_j \rangle = x_i^T x_j$$

Properties

1) G is PSD : $u^T G u \geq 0$ for any u

(All eigenvalues of G being non-negative)

2) G is symmetric: $G_{ij} = G_{ji}$

Kernel examples

$$K(x_i, x_j) = \phi(x_i)^T \phi(x_j)$$

$$\phi(x_i) = [1, \sqrt{2}\, x_1, \sqrt{2}\, x_2, x_1 x_2, x_1^2, x_2^2] \in \mathbb{R}^6$$

$$\phi(x_i)^T \phi(x_j) = 1 + 2 x_i(1) x_j(1) + 2 x_i(2) x_j(2) +$$

$$+ 2 x_i(1) x_j(1) x_i(2) x_j(2) + x_i(1)^2 x_j(1)^2$$

$$+ x_i(2) x_j(2)^2 = (1 + x_i^T x_j)^2$$

$$K[i,j] = (1 + x_i^T x_j)^2$$

This kernel is for ϕ that is a polynomial of degree 2.

1) Polynomials of higher degree: $d > 2$

$$K_{ij} = K[i,j] = (1 + x_i^T x_j)^d$$

2) Sigmoid kernel

$$K(x_i, x_j) = \tanh(\alpha x_i^T x_j + \beta)$$

3) Gaussian / RBF kernel

$$k(x_i, x_j) = \exp\{-\|x_i - x_j\|^2 / 2\sigma^2\}$$

Theorem

A function $k(x_i, x_j)$ is a valid kernel if

1) k is symmetric: $k(x_i, x_j) = k(x_j, x_i)$ ← this is easy

2) k is PSD ← harder to check

$$\sum_i \sum_j y_i y_j k(x_i, x_j) \geq 0 \quad \forall y \in \mathbb{R}^n$$

Mercer's Theorem

A function $k(x_i, x_j)$ is PSD iff it can be expressed as an inner product:

$$k(x_i, x_j) = \langle \phi(x_i), \phi(x_j) \rangle = \phi(x_i)^T \phi(x_j)$$

Let's look at the Gaussi kernel

$$\exp\left\{-\frac{\|x_i - x_j\|^2}{2\sigma^2}\right\} = \exp\left\{-\frac{\|x_i\|^2}{2\sigma^2}\right\} \cdot \exp\left\{-\frac{\|x_j\|^2}{2\sigma^2}\right\} \cdot \exp\left\{\frac{-2x_i^T x_j}{2\sigma^2}\right\}$$

$$= \exp\left\{\frac{-\|x_i\|^2}{2\sigma^2}\right\} \exp\left\{-\frac{\|x_j\|^2}{2\sigma^2}\right\} \underbrace{\sum_{k=0}^{\infty} \frac{1}{k!} (x_i^T x_j)^k / \sigma^2}_{e^x = \sum_{n=0}^{\infty} \frac{x^n}{n!} \text{ (Taylor)}}$$

Note that the high dim feature space here is infinite-dimensional

Infinite sum of PSD is still PSD

Proof of Mercer's Theorem (high level)

1) We assume K is symmetric, therefore we can apply spectral theorem

$$K = V \Lambda V^T$$

2) If K is PSD then $\lambda_i \geq 0$ (eigenvalues)

3) We can define a mapping

$$\phi(x_j) = [\sqrt{\lambda_1}\, V_1(j), \sqrt{\lambda_2}\, V_2(j), \dots, \sqrt{\lambda_n}\, V_n(j)]^T$$

Then by construction we get

$$K(i,j) = \langle \phi(x_i), \phi(x_j) \rangle$$

Gaussian Kernel and bandwidth selection

$$K(x_i, x_j) = \exp\{-\|x_i - x_j\|^2 / 2\sigma^2\}$$

σ is a bandwidth

1) If $\sigma \gg \|x_i - x_j\| \quad \forall x_i, x_j$

$$\frac{\|x_i - x_j\|}{\sigma} \longrightarrow 0 \qquad K(x_i, x_j) \approx 1$$

2) If $\sigma \ll \|x_i - x_j\|$

$$\frac{\|x_i - x_j\|}{\sigma} \rightarrow \infty \qquad k(x_i, x_j) \approx 0$$

3) By correct choice of σ, the Gaussian kernel preserves locality:

we'll have high k_{ij} for x_i, x_j that are similar

and $k_{ij} \approx 0$ for x_i, x_j that are distant

4) Typical choice $\sigma = \underset{x_i, x_j \in X}{\text{median}} \{\|x_i - x_j\|\}$

5) small bandwidth reduces bias - average only over points that are similar to one another
but it also increases the variance in noisy data

Nystrom / Out-of-sample extension

The kernel has size $n \times n$

$O(n^2)$ to calculate
$O(n^3)$ to calculate eigenvectors } naive estimate

Can reduce computational complexity by approximation

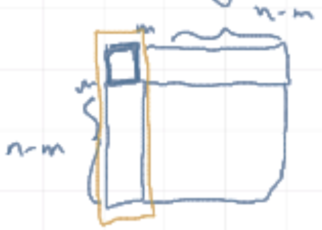
Assumption of Nyström approach: kernel is low-rank (rank $d \ll n$)

Sample m random points from the data

Calculation will rely on computing K only between the m points and the n points

$K = \begin{bmatrix} K_{11} & K_{12} \\ K_{21} & K_{22} \end{bmatrix}$ $\qquad K_{12} = K_{21}^T$ $\qquad \measuredangle$

$\Downarrow$



K_{11} $\quad m \times m$

K_{21} $\quad (n-m) \times m$

K_{22} $\quad (n-m) \times (n-m)$

For low rank approximation

$K = U_d \Lambda_d U_d^T \quad \leftarrow$

where $U = \begin{bmatrix} U_1 \\ U_2 \end{bmatrix} \in \mathbb{R}^{n \times d}$

Λ - diagonal $d \times d$ matrix

We're going to take an EVD of K_{11} $(O(m^3))$

$K_{11} = U_1 \Lambda U_1^T$ $\qquad \underbrace{\begin{bmatrix} U_1 \\ U_2 \end{bmatrix}}_{d \text{ eigenvectors}} [\Lambda_d] [U_1 \ U_2]^T$

$K_{21} = U_2 \Lambda U_1^T$

$K_{21} U_1 = U_2 \Lambda$

$\underline{K_{21}} \, \underline{U_1} \Lambda^{-1} = \underline{U_2} \quad \leftarrow$

$K_{22} = U_2 \Lambda U_2^T$

Summary

1) PCA can't "identify" non linear structures

2) Kernel PCA - map data to high dim space

Apply PCA there

We avoid having to calculate R_ϕ $(l \times l)$

By calculating α Eigenvectors of $K_{ij} = \phi(x_i)^T \phi(x_j)$

3) $\underbrace{\phi(x_i)^T}_{\text{don't have to calculate}} v = [K\alpha]_i$

4) All properties of PCA (max variance, minimum reconstruction error, uncorrelated features) hold in the <u>high-dim</u> space.

5) Kernel PCA just relies on eigenvalue decomp

6) Flexible: can use different kernels

Limitations

1) Reconstruction? need pre-image solution

2) Interpretability