Chapter 4: Kernel Principal Component Analysis[a]

PCA (Chapter 1) tries to find a low-dimensional linear subspace on which the data points $\{x_i^0\}_{i=1}^n$ can be well represented. However, in some cases it is not possible to find such a linear subspace.

To illustrate this point let $\{x_i^0\}_{i=1}^n$ be $n = 300$ observations of dimension $p = 3$ sampled i.i.d. using

$$\begin{aligned} X_{i1}^0 &= (1+2B_i)\cos(\phi_i)\sin(\theta_i) + \epsilon_{i1}, \quad && \epsilon_{i1} \sim \mathcal{N}_1(0, 0.3^2) \\ X_{i2}^0 &= (1+2B_i)\sin(\phi_i)\sin(\theta_i) + \epsilon_{i2}, \quad && \epsilon_{i2} \sim \mathcal{N}_1(0, 0.3^2) \\ X_{i3}^0 &= (1+2B_i)\cos(\theta_i) + \epsilon_{i3}, \quad && \epsilon_{i3} \sim \mathcal{N}_1(0, 0.3^2) \end{aligned} \tag{4.1}$$

where $\phi_i \sim \mathcal{U}(0, 2\pi)$, $\theta_i \sim \mathcal{U}(0, \pi)$ and $B_i \sim \text{Bernoulli}(0.5)$.

The 2-dimensional reduction $\{x_i'\}_{i=1}^n$ of $\{x_i^0\}_{i=1}^n$ obtained with PCA is shown in Figure 4.1. From this figure we observe (i) that from $\{x_i'\}_{i=1}^n$ we cannot identify the existence of two groups of observations and (ii) that, compared to sub-dataset $\{(x_{i1}, x_{i2})\}_{i=1}^n$, the reduction $\{x_i'\}_{i=1}^n$ of the observations does not provide any additional useful information.

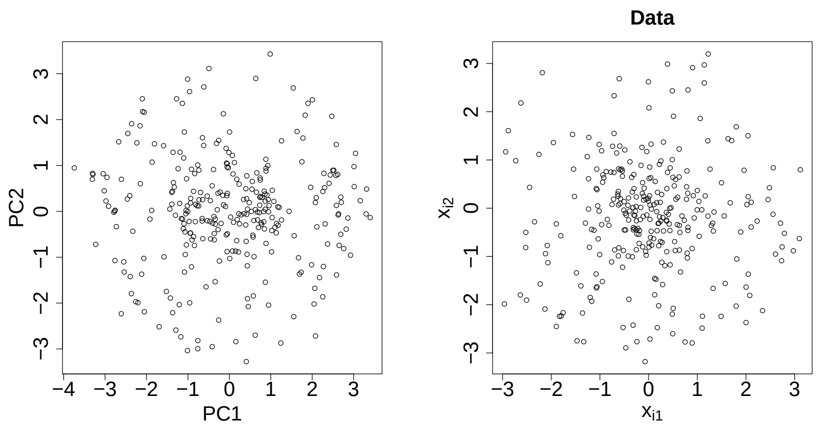

Figure 4.1: Two dimensional reduction of $\{x_i^0\}_{i=1}^n$ obtained with PCA (left) and observations $\{(x_{i1}, x_{i2})\}_{i=1}^n$ (right).

[a]The main reference for this chapter is [1, Section 12.3].

Reason why PCA fails in the previous example

To understand the problem with PCA it is useful to look at the observations $\{x_i^0\}_{i=1}^n$.

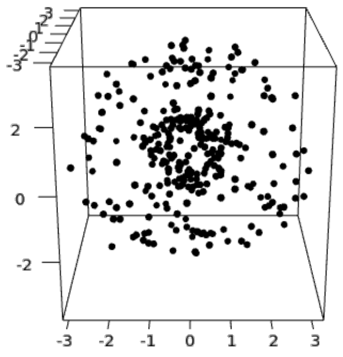

Figure 4.2: Observations $\{x_i^0\}_{i=1}^n$ for the introductory example.

From Figure 4.2, it is clear that for this dataset the problem PCA encounters is to determine the axis (i.e. the component) of maximum variance.

Indeed, and informally speaking, due to the spherical nature of the data every direction is equally important in term of variance.

In addition, we see in Figure 4.2 that the two groups of observations cannot be separated in $\mathbb{R}^3$ with an affine hyperplane, that is, with a 2-dimensional linear sub-space.

For this reason, the linear subspace spanned by the first two eigenvectors of the covariance matrix cannot separate the two groups of observations.

Continuation of the introductory example

It is known that projecting the observations in a higher dimensional space usually makes them more easily linearly separable[a].

This suggests the following approach to obtain a two dimensional representation of the data which is more informative than the one obtained in Figure 4.1 with PCA:

1. Map the observations $\{x_i^0\}_{i=1}^n$ into a space of dimension $m \gg p$, using a mapping $\Phi : \mathbb{R}^p \to \mathbb{R}^m$.
2. Use PCA to compute the two dimensional reduction $\{\phi_i'\}_{i=1}^n$ of the transformed observations $\{\phi_i^0 := \Phi(x_i^0)\}_{i=1}^n$.

Figure 4.3 shows the two dimensional reduction $\{\phi_i'\}_{i=1}^n$ obtained for a mapping $\Phi : \mathbb{R}^p \to \mathbb{R}^m$ such that $m = \infty$[b].

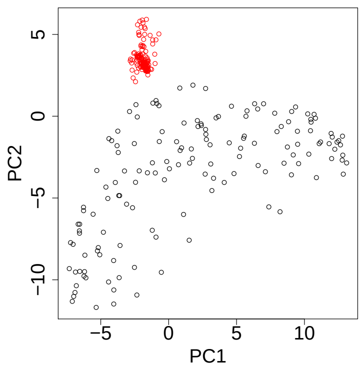

Figure 4.3: Two dimension reduction of $\{\phi_i^0\}_{i=1}^n$ obtained with PCA.

We observe from Figure 4.3 that the approach described above, called kernel PCA (for reasons that will become obvious later) successfully allows to have a two dimensional representation of the data that separates the two groups of observations.

[a]Cover's theorem is sometimes used to support this claim.

[b]Here Φ is the feature map associated to the Gaussian kernel $k(x, x') = \exp(-\|x - x'\|^2)$.

Kernel PCA: Set-up and notation

Below we let $\Phi : \mathbb{R}^p \to \mathcal{H}$ be such that

- $\mathcal{H} \subseteq \mathbb{R}^m$ for some $m \in \mathbb{N} \cup \{\infty\}$,
- $\Phi(x)^\top \Phi(x') < \infty$ for all $x, x' \in \mathbb{R}^p$.

Remark: The dimension of $\mathcal{H}$ can therefore be infinite, and in this case $\Phi(x)^\top \Phi(x')$ is assumed to exist pointwise.

Next, we let $\phi_i^0 = \Phi(x_i^0)$ for all i and consider the problem of using PCA to reduce the dimension of the transformed observations $\{\phi_i^0\}_{i=1}^n$.

A naive approach to do so consists in computing the transformed data matrix $\mathbf{\Phi}^0$ and then computing the PCs following the approach introduced in Chapter 1. However, doing this is computationally expensive, since this requires to compute $\phi_i^0 \in \mathcal{H}$ where, typically, $\mathcal{H}$ is a high (and usually infinite) dimensional space.

As we will see in this chapter, it turns out that if we are able to evaluate pointwise the function $k : \mathbb{R}^p \times \mathbb{R}^p \to \mathbb{R}$ defined by

$$k(x, x') = \Phi(x)^\top \Phi(x'), \quad (x, x') \in \mathbb{R}^p \times \mathbb{R}^p,$$

then PCA can be applied on the transformed data set without having to explicitly compute the transformed observations $\{\phi_i^0\}_{i=1}^n$.

Preliminaries

Let $\mathbf{\Phi}^0 = [\phi_{ij}^0]$ and $\boldsymbol{K}^0 = \mathbf{\Phi}^0\big(\mathbf{\Phi}^0\big)^\top = [k(x_i^0, x_l^0)]$.

Next, for all i and j let $\phi_{ij} = \phi_{ij}^0 - \frac{1}{n}\sum_{l=1}^n \phi_{lj}^0$, so that $\mathbf{\Phi} := [\phi_{ij}]$ is the matrix containing the centred transformed observations, $\boldsymbol{K} = \mathbf{\Phi}\mathbf{\Phi}^\top$.

In addition, for all $x, x' \in \mathbb{R}^p$ let

$$\begin{aligned}\tilde{k}(x,x') = k(x,x') &- \frac{1}{n}\sum_{s=1}^n k(x, x_s^0) - \frac{1}{n}\sum_{s=1}^n k(x_s^0, x') \\ &+ \frac{1}{n^2}\sum_{s=1}^n\sum_{s'=1}^n k(x_s^0, x_{s'}^0).\end{aligned} \tag{4.2}$$

As shown in the following lemma, the matrix $\boldsymbol{K}$ has $\tilde{k}(x_i^0, x_l^0)$ as element (i,l) and thus being able to evaluate $k(\cdot,\cdot)$ is enough to compute the Gram matrix $\boldsymbol{K}$. In particular, we can compute $\boldsymbol{K}$ without having to actually center the transformed observations $\{\phi_i^0\}_{i=1}^n$.

Lemma 4.1 *We have* $\boldsymbol{K} = \boldsymbol{C}_n\boldsymbol{K}^0\boldsymbol{C}_n = [\tilde{k}(x_i^0, x_l^0)]$.

Proof: For all $i, l \in \{1, \dots, n\}$ we have

$$\begin{aligned}\phi_i^\top\phi_l &= \Big(\phi_i^0 - \frac{1}{n}\sum_{s=1}^n \phi_s^0\Big)^\top\Big(\phi_l^0 - \frac{1}{n}\sum_{s=1}^n \phi_s^0\Big) \\ &= \big(\phi_i^0\big)^\top\phi_l^0 - \frac{1}{n}\sum_{s=1}^n \big(\phi_i^0\big)^\top\phi_s - \frac{1}{n}\sum_{s=1}^n \big(\phi_s^0\big)^\top\phi_l + \frac{1}{n^2}\sum_{s=1}^n\sum_{s'=1}^n \big(\phi_s^0\big)^\top\phi_{s'}^0 \\ &= k(x_i^0, x_l^0) - \frac{1}{n}\sum_{s=1}^n k(x_i^0, x_s^0) - \frac{1}{n}\sum_{s=1}^n k(x_s^0, x_l^0) + \frac{1}{n^2}\sum_{s=1}^n\sum_{s'=1}^n k(x_s^0, x_{s'}^0) \\ &= \tilde{k}(x_i^0, x_l^0)\end{aligned}$$

and thus

$$\boldsymbol{K} = \boldsymbol{K}^0 - \big(\frac{1}{n}\mathbf{1}_n\mathbf{1}_n^\top\big)\boldsymbol{K}^0 - \boldsymbol{K}^0\big(\frac{1}{n}\mathbf{1}_n\mathbf{1}_n^\top\big) + \big(\frac{1}{n}\mathbf{1}_n\mathbf{1}_n^\top\big)\boldsymbol{K}^0\big(\frac{1}{n}\mathbf{1}_n\mathbf{1}_n^\top\big) = \boldsymbol{C}_n\boldsymbol{K}^0\boldsymbol{C}_n.$$

The proof is complete.

Eigenvalue problems in $\mathcal{H}$

Let $\tilde{\lambda}_1 \geq \cdots \geq \tilde{\lambda}_n \geq 0$ be the eigenvalues of $\boldsymbol{K}$ and $\tilde{\boldsymbol{A}} \in O(n)$ be the matrix having the corresponding eigenvectors as columns.

Let $\boldsymbol{S}_\Phi = \frac{1}{n}\boldsymbol{\Phi}^\top\boldsymbol{\Phi}$ and note that, by Theorem 1.2,

$$r := \operatorname{rank}(\boldsymbol{S}_\Phi) = \operatorname{rank}(\boldsymbol{K}) \leq \min(n, m).$$

Let $\lambda_{\Phi,1} \geq \cdots \geq \lambda_{\Phi,r} > 0$ be the r non-zero eigenvalues of $\boldsymbol{S}_\Phi$ and $\boldsymbol{A}_\Phi = [a_{(\Phi,1)} \dots a_{(\Phi,r)}] \in O(r)$ be the matrix having the corresponding eigenvectors as columns.

Then, by Theorem 1.2, all the positive eigenvalues of $\boldsymbol{S}_\Phi$ and the associated matrix $\boldsymbol{A}_\Phi$ can be computed from the Gram matrix $\boldsymbol{K}$, as stated in the following result.

Corollary 4.1 *We have*

$$\lambda_{\Phi,j} = \frac{\tilde{\lambda}_j}{n}, \quad a_{(\Phi,j)} = \frac{\boldsymbol{\Phi}^\top \tilde{a}_{(j)}}{\tilde{\lambda}_j^{1/2}}, \quad \forall j \in \{1, \dots, r\}.$$

This result is crucial for KPCA to be implementable in practice, and for this reason we prove it from scratch in what follows.

Proof of Corollary 4.1

Let $\lambda_\Phi > 0$ be a strictly positive eigenvalue of $\boldsymbol{S}_\Phi$ and $v_\Phi \in \mathcal{H}$ be a corresponding unitary eigenvector. Then,

$$\lambda_\Phi v_\Phi = \boldsymbol{S}_\Phi v_\Phi = \Big(\frac{1}{n}\sum_{i=1}^{n} \phi_i \phi_i^\top\Big) v_\Phi = \frac{1}{n}\sum_{i=1}^{n} \phi_i(\phi_i^\top v_\Phi) \tag{4.3}$$

or, equivalently,

$$v_\Phi = \frac{1}{n\lambda_\Phi}\sum_{i=1}^{n}(\phi_i^\top v_\Phi)\phi_i = \sum_{l=1}^{n} w_l \phi_l, \quad \text{where } w_l = \frac{\phi_l^\top v_\Phi}{n\lambda_\Phi}. \tag{4.4}$$

Remark: Computing the eigenvector $v_\Phi \in \mathcal{H}$ is therefore equivalent to computing the vector $w = (w_1, \ldots, w_n) \in \mathbb{R}^n$.

Substituting the expression (4.4) for v_Φ into (4.3) yields

$$\lambda_\Phi \sum_{l=1}^{n} w_l \phi_l = \frac{1}{n}\sum_{i=1}^{n} \phi_i \phi_i^\top \sum_{l=1}^{n} w_l \phi_l \tag{4.5}$$

showing that, for all $s \in \{1, \ldots, n\}$,

$$\begin{aligned} n\lambda_\Phi \sum_{l=1}^{n} w_l \tilde{k}(x_s^0, x_l^0) &= n\lambda_\Phi \sum_{l=1}^{n} w_l \phi_s^\top \phi_l \\ &= \sum_{i=1}^{n} (\phi_s^\top \phi_i) \sum_{l=1}^{n} w_l \phi_i^\top \phi_l \\ &= \sum_{i=1}^{n} \tilde{k}(x_s^0, x_i^0) \sum_{l=1}^{n} w_l \tilde{k}(x_i^0, x_l^0). \end{aligned} \tag{4.6}$$

Proof of Corollary 4.1 (end)

Writing (4.6) in matrix form implies that λ_Φ and w verifies

$$\boldsymbol{K}^2 w = n\lambda_\Phi \boldsymbol{K} w \tag{4.7}$$

showing that the pair $(\tilde{\lambda}, \tilde{v}) := (n\lambda_\Phi, \boldsymbol{K}w)$ is such that $\boldsymbol{K}\tilde{v} = \tilde{\lambda}\tilde{v}$, that is, that $n\lambda_\Phi$ is an eigenvalue of $\boldsymbol{K}$.

Since the above computations hold for an arbitrary strictly positive eigenvalue of $\boldsymbol{S}_\Phi$, and since $\text{rank}(\boldsymbol{S}_\Phi) = \text{rank}(\boldsymbol{K}) = r$, it follows that $\lambda_{\Phi,j} = \tilde{\lambda}_j/n$ for $j = 1, \dots r$.

To complete the proof it remains to compute the eigenvector v_Φ associated to $\lambda_\Phi \in \{\lambda_{\Phi,1}, \dots, \lambda_{\Phi,r}\}$ which, by (4.4), requires to first find a $w \in \mathbb{R}^n$ such that (4.7) holds.

To this aim, remark that since (4.7) is equivalent to

$$\boldsymbol{K}(\boldsymbol{K}w - \lambda_\Phi n w) = 0,$$

it follows (4.7) holds for $w = c\tilde{v}$, where $\tilde{v}$ is the orthonormal eigenvector of $\boldsymbol{K}$ associated to the eigenvalue $\tilde{\lambda} := n\lambda_\Phi$ and where $c > 0$ is such that $\|v_\Phi\| = 1$.

More precisely, the constant c must be such that

$$\begin{aligned} 1 &= \Big\| \sum_{l=1}^n w_l \phi_l \Big\|^2 = \sum_{l'=1}^n \sum_{l=1}^n w_l w_{l'} \phi_l^\top \phi_{l'} = \sum_{l'=1}^n \sum_{l=1}^n w_l w_{l'} \tilde{k}(x_l^0, x_{l'}^0) \\ &= w^\top \boldsymbol{K} w = c^2\, \tilde{v}^\top \big(\boldsymbol{K}\tilde{v}\big) = c^2 \tilde{v}^\top \big(\tilde{\lambda}\tilde{v}\big) = c^2 \tilde{\lambda} \tilde{v}^\top \tilde{v} = c^2 \tilde{\lambda} \end{aligned}$$

and thus $c = \tilde{\lambda}^{-1/2}$. The proof is complete. □

Principal components extraction

With $\tilde{k}(\cdot,\cdot)$ as defined in (4.2), let

$$\tilde{k}_n(x) = (\tilde{k}(x_1^0, x), \dots, \tilde{k}(x_n^0, x)), \quad x \in \mathbb{R}^p$$

and $\tilde{y}_{(j)}$ be the jth PC associated to the data matrix $\mathbf{\Phi}$.

Then, as in Chapter 1 and using Corollary 4.1,

$$\begin{aligned}\tilde{y}_{ij} = a_{(\Phi,j)}^\top \phi_i = \phi_i^\top a_{(\Phi,j)} &= \tilde{\lambda}_j^{-1/2} \phi_i^\top \mathbf{\Phi}^\top \tilde{a}_{(j)} \\ &= \tilde{\lambda}_j^{-1/2} \tilde{k}_n(x_i^0)^\top \tilde{a}_{(j)} \\ &= \tilde{\lambda}_j^{-1/2} \tilde{a}_{(j)}^\top \tilde{k}_n(x_i^0).\end{aligned}$$

Letting $\tilde{\mathbf{\Lambda}} = \mathrm{diag}(\tilde{\lambda}_1, \dots, \tilde{\lambda}_r)$ and $\check{\boldsymbol{A}} = \tilde{\boldsymbol{A}}\tilde{\mathbf{\Lambda}}^{-1/2} \in O(r)$, it follows that the matrix $\tilde{\boldsymbol{Y}} = [\tilde{y}_{ij}] \in \mathbb{R}^{n\times r}$ containing the PCs associated to the non-zero eigenvalues of $\boldsymbol{S}_\Phi$ is given by

$$\tilde{\boldsymbol{Y}} = \boldsymbol{K}\check{\boldsymbol{A}}.$$

Key remark: The computation of the PCs only requires to evaluate $k(\cdot,\cdot)$ (kernel trick).

More generally, for $q \leq r$ the coordinate of the projection of a point $x \in \mathbb{R}^p$ onto span$\{a_{(\Phi,j)}, j = 1, \dots, r\}$ is given by

$$\tilde{y}_n^{(q)}(x) = \check{\boldsymbol{A}}_{1:q}^\top \tilde{k}_n(x). \tag{4.8}$$

Remark: The function $x \mapsto \tilde{y}_n^{(q)}(x)$ is non-linear, making KPCA with $q < p$ a non-linear dimension reduction method.

Remark: We recover standard PCA when $\Phi(x) = x$, that is, when $k(x, x') = x^\top x'$.

Choice of the feature map Φ

For Kernel PCA to be implementable the mapping $\Phi : \mathbb{R}^p \to \mathbb{R}^m$ must be such that the the function $k(\cdot,\cdot) = \Phi^\top(\cdot)\Phi(\cdot)$ can be (easily) evaluated pointwise.

In practice, we proceed the over way round: we choose a function $k : \mathbb{R}^p \times \mathbb{R}^p \to \mathbb{R}$ which is

(i) easy to evaluate,

(ii) for which we know that, for some $m \in \mathbb{N} \cup \{\infty\}$, there exists a feature map $\Phi : \mathbb{R}^p \to \mathbb{R}^m$ such that $k(x,x') = \Phi(x)^\top \Phi(x')$ for all $x, x' \in \mathbb{R}^p$.

Typically (see Theorem 4.1 below) the above condition (ii) is satisfied if $k : \mathbb{R}^p \times \mathbb{R}^p \to \mathbb{R}$ is a kernel function (hence the name kernel PCA!).

Definition 4.2 *A function $k : \mathbb{R}^p \times \mathbb{R}^p \to \mathbb{R}$ is called a kernel if it is symmetric and positive semi-definite, i.e. if*

$$k(x_1', x_2') = k(x_2', x_1'), \quad \sum_{i,j=1}^{m} a_i a_j k(x_i', x_j') \geq 0$$

for all $(a_1, \dots, a_m) \in \mathbb{R}^m$, all $(x_1', \dots, x_m') \in \mathcal{X}^m$ and all $m \in \mathbb{N}$.

Remark that if k is a kernel then the matrix $\boldsymbol{K} = [k(x_i^0, x_l^0)]$ is symmetric and positive semi-definite. Consequently, when k is a kernel even if the above condition (ii) does not hold the values $\{\tilde{\lambda}_{\Phi,j}\}_{i=1}^{r}$ defined in Corollary 4.1 are strictly positive.

Mercer's theorem

Mercer's theorem[a] provides a sufficient condition for a kernel k to satisfy the above condition (ii).

Theorem 4.1 (Mercer's theorem) *Let $\mathcal{X} \subset \mathbb{R}^p$ be a compact set an k be a continuous kernel on $\mathcal{X}$. Then, there exist real-valued continuous functions $(e_j)_{j\geq 1}$ and non-negative real numbers $\lambda_1 \geq \lambda_2 \geq \dots$ such that*[b]

$$k(x,x') = \sum_{j=1}^{\infty} \lambda_j e_j(x) e_j(x'), \quad \forall x, x' \in \mathcal{X}. \tag{4.9}$$

Therefore, if k and $\mathcal{X} \subset \mathbb{R}^p$ are as in Theorem 4.1 then, with $m = \infty$, $k(\cdot,\cdot) = \Phi(\cdot)^\top \Phi(\cdot)$ where $\Phi : \mathcal{X} \times \mathcal{X} \to \mathbb{R}^m$ is defined by

$$\Phi(x) = \Big(\lambda_1^{1/2} e_1(x), \lambda_2^{1/2} e_2(x), \dots\Big).$$

Remark: In general, for a given kernel k as in the theorem the corresponding definition of the feature map Φ is unknown.

Remark: The assumption that k is defined on a compact set $\mathcal{X} \subset \mathbb{R}^p$ is not a severe limitation. Indeed, if the observations $\{x_i^0\}_{i=1}^n$ are in $\mathbb{R}^p$ and k' is a kernel on $\mathcal{X} := [0,1]^p$ that satisfies the assumptions of the theorem then the kernel $k : \mathbb{R}^p \times \mathbb{R}^p \to \mathbb{R}$ defined by $k(x,x') = k'(f(x), f(x'))$, where $f : \mathbb{R}^p \to (0,1)^p$ is one-to-one an continuous (e.g. f is the component-wise logistic transformation), satisfies the above condition (ii).

[a]See [5], Theorem 11.15

[b]The equality holds pointwise in (4.9).

Two popular kernels

- The Gaussian kernel $k_\gamma(x,x') = \exp(-\|x-x'\|^2/\gamma^2)$ is a popular choice of kernel for KPCA.

  For this kernel it can be shown that for $p=1$ the feature map Φ is defined by

$$\Phi(x) = e^{-\frac{x^2}{\gamma^2}}\Big(1, \sqrt{\frac{(2/\gamma^2)}{1!}}\,x, \sqrt{\frac{(2/\gamma^2)^2}{2!}}\,x^2, \sqrt{\frac{(2/\gamma^2)^3}{3!}}\,x^3, \dots\Big).$$

- An other popular choice for k is the Matérn kernel $k_{\alpha,\gamma}$, defined by

$$k_{\alpha,\gamma}(x,x') = \frac{2^{1-\alpha}}{\Gamma(\alpha)}\Big(\frac{\sqrt{2\alpha}\|x-x'\|}{\gamma}\Big)^{\alpha} K_\alpha\Big(\frac{\sqrt{2\alpha}\|x-x'\|}{\gamma}\Big),$$

  with K_α the modified Bessel function of the 2nd kind of order α.

  Some particular cases:

$$k_{1/2,\gamma}(x,x') = \exp\Big(-\frac{\|x-x'\|}{\gamma}\Big),$$ [exponential kernel]

$$k_{3/2,\gamma}(x,x') = \Big(1+\frac{\sqrt{3}\|x-x'\|}{\gamma}\Big)\exp\Big(-\frac{\|x-x'\|}{\gamma}\Big)$$

$$k_{5/2,\gamma}(x,x') = \Big(1+\frac{\sqrt{3}\|x-x'\|}{\gamma}+\frac{5\|x-x'\|^2}{3\gamma^2}\Big)\exp\Big(-\frac{\|x-x'\|}{\gamma}\Big).$$

Practicalities

- KPCA depends heavily on the choice of the kernel k. In particular, if k is poorly chosen then it may be the case that all the eigenvalues $\lambda_1, \dots, \lambda_r$ are approximatively equal (in which case kernel PCA will not provide any useful information).

- In practice we often let $k(x, x') = f(\|x - x'\|/c)$, in which case KPCA heavily depends on the choice for $c > 0$. A standard default approach is to let $c = \sqrt{h_n/2}$ with h_n a median of the set $\{\|x_i^0 - x_l^0\|^2\}_{i,l=1}^n$ (median heuristic).

- As for standard PCA, it is important that the (original) observations are on an appropriate scale before performing KPCA.

  Indeed, in practice k is often such that $k(x_i^0, x_l^0) = f(\|x_i^0 - x_l^0\|)$ for some function $f : \mathbb{R} \to \mathbb{R}$, and therefore for the distance $\|x_i^0 - x_l^0\|$ to be meaningful all the variables must be on a similar scale.

  Remark: When $k(x, x') = f(\|x - x'\|)$ applying KPCA on $\boldsymbol{X}^0$ or on the centred data matrix $\boldsymbol{X}$ is equivalent.

  It is useful to note that in KPCA rescaling the data matrix $\boldsymbol{X}^0$ is equivalent to modifying the definition of the kernel k.

  Indeed, if $\boldsymbol{D}$ is the scaling matrix (i.e. $\boldsymbol{D}$ is a diagonal matrix with strictly positive diagonal entries) then applying KPCA on $\tilde{\boldsymbol{X}}^0 := \boldsymbol{X}^0\boldsymbol{D}$ using the kernel $k(x, x')$ is equivalent to applying KPCA on $\boldsymbol{X}^0$ using the kernel $k'(x, x') = k(\boldsymbol{D}x, \boldsymbol{D}x')$.

Practicalities

KPCA requires to compute the eigen-decomposition of the matrix $\boldsymbol{K} \in \mathbb{R}^{n\times n}$, which requires $\mathcal{O}(n^3)$ operations and a storage complexity of size $\mathcal{O}(n^2)$. Consequently, even for a moderate sample size n approximation methods are needed to use kernel PCA in practice.

The Nyström method (also called low-rank approximation method) is a standard way to reduce the computational cost of KPCA (and of most kernel methods).

Given and integer $d < n$, Nyström method amounts to approximating k by a kernel $\tilde{k}^{(d)}$ such that the matrix $\tilde{\boldsymbol{K}}^{(d)}$, obtained by replacing k by $\tilde{k}^{(d)}$ in the definition of $\boldsymbol{K}$, has rank at most d.

Assuming that k can be written as in (4.9), a natural idea is to take for $\tilde{k}^{(d)}$ the kernel obtained by keeping in the expansion (4.9) of k only the terms associated to the d largest values of $(\lambda_j)_{j\geq 1}$, that is to take $\tilde{k}^{(d)} = k^{(d)}$ with $k^{(d)}$ defined by

$$k^{(d)}(x, x') = \sum_{j=1}^{d} \lambda_j e_j(x) e_j(x').$$

The real numbers $\{\lambda_j\}_{j=1}^d$ and functions $\{e_j\}_{j=1}^d$ are typically unknown, and the idea of Nyström method is to use the data $\{x_i^0\}_{i=1}^n$ to estimate these quantities, and then to take for $\tilde{k}^{(d)}$ the resulting approximation of $k^{(d)}$.

Practicalities (end)

More precisely, in the Nyström method the approximation $\tilde{k}^{(d)}$ of k is defined by

$$\tilde{k}^{(d)}(x, x') = k_d(x)^\top (\boldsymbol{K}_d^0)^{-1} k_d(x'), \quad x, x' \in \mathbb{R}^p \tag{4.10}$$

where $\boldsymbol{K}_d^0$ denotes the $d \times d$ matrix containing the first d rows and columns of $\boldsymbol{K}^0 = [k(x_i, x_l)]$ and where

$$k_d(x) = \big(k(x_1^0, x), \dots, k(x_d^0, x)\big), \quad \forall x \in \mathbb{R}^p.$$

Remark: This approximation $\tilde{k}^{(d)}$ of $k^{(d)}$ will be justified in Chapter 12. In that chapter we will also see that if $\tilde{k}^{(p)}$ is as defined in (4.10) then the rank of the $n \times n$ matrix $\tilde{\boldsymbol{K}}^{(d)}$ is indeed at most d.

The rank of $\tilde{\boldsymbol{K}}^{(d)}$ being a most d, it can shown that the storage and complexity requirements of computing the eigen-decomposition of this matrix is $\mathcal{O}(nd)$ and $\mathcal{O}(nd^2)$, respectively. Hence, as d decreases KPCA becomes faster but, on the other hand, the approximation $\tilde{k}^{(d)}$ of k becomes less accurate.

Remark: The definition (4.10) of $\tilde{k}^{(d)}$ depends on the order of the observations, and thus different permutations of the data $\{x_i^0\}_{i=1}^n$ lead to different approximations $\tilde{k}^{(d)}$ of k.

Application of KPCA: Non-linear classification using logistic regression models

This application of KPCA is introduced through an example.

Let $\{x_i^0\}_{i=1}^n$ be $n = 300$ bivariate observations sampled using

$$X_{i1}^0 = R_i \cos(\theta_i) + \epsilon_{i1}, \quad X_{i2}^0 = R_i \sin(\theta_i) + \epsilon_{i2}, \quad \epsilon_i \overset{\text{iid}}{\sim} \mathcal{N}_2(0, 0.5^2 \boldsymbol{I}_2)$$

where $\theta_i \overset{\text{iid}}{\sim} \mathcal{U}(0, 2\pi)$ and $\mathbb{P}(R_i = 1) = \mathbb{P}(R_i = 2) = 1/2$.

These observations are represented in Figure 4.4 below.

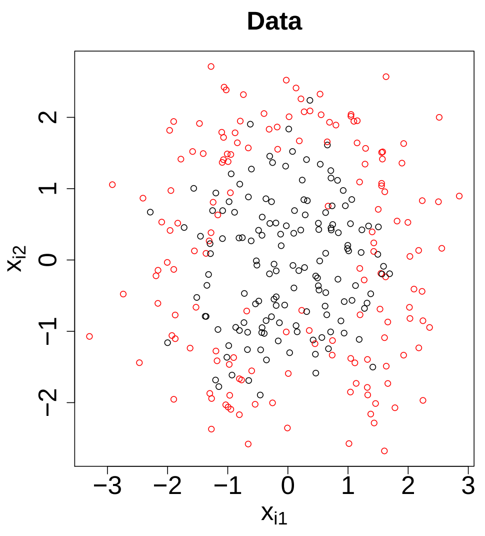

Figure 4.4: Observations $\{x_i^0\}_{i=1}^n$

We now let $z_i = 1$ if $r_i = 2$ and $z_i = 0$ otherwise, and consider the problem of using a logistic regression model to predict the value of z_i.

To do so, we will fit a logistic regression model to estimate $\mathbb{P}(Z = 1; x)$ for a given $x \in \mathbb{R}^2$, and then make the prediction $\hat{z}(x) = 1$ if the estimated probability is larger than 0.5 and $\hat{z}(x) = 0$ otherwise.

The estimated model is tested on a test set of size 300, generated as the training set.

Logistic regression on the original data points

We start by fitting logistic regression using the original data points $\{x_i^0\}_{i=1}^n$ as covariates.

In this case, for $x \in \mathbb{R}^2$ the model assumes that

$$\mathbb{P}(Z=1;x) \in \Big\{\mathbb{P}_{(\alpha,\beta)}(Z_i=1;x) := \frac{1}{1+e^{-\alpha-\beta^\top x}},\ (\alpha,\beta)\in\mathbb{R}^3\Big\}. \tag{4.11}$$

It is clear that the decision boundary of the classifier obtained using a logistic regression model is linear. However, from Figure 4.4, we clearly see that the two groups of observations cannot be linearly separated and, consequently, we expect that the model (4.11) will perform poorly in term of predictions[a].

The percentage of correct prediction of the resulting model on the test set is only 51.67%. Therefore, for this example the predictions obtained with the model (4.11) are not better than those obtained by flipping a coin.

[a]More precisely, for a given value of $x_1 \in \mathbb{R}$ (resp. of $x_2 \in \mathbb{R}$) and of (α,β) the model (4.11) assumes that $\mathbb{P}_{(\alpha,\beta)}(Z=1;(x_1,x_2))$ is a monotone function of x_2 (resp. of x_1), an assumption which is clearly not suitable for the data that we consider (see Figure 4.4).

Using KPCA within logistic regression

As mentioned at the beginning of this chapter, for $\Phi : \mathbb{R}^p \to \mathbb{R}^m$ with $m \gg p$ (possibly $m = \infty$) the transformed observations $\{\Phi(x_i^0)\}_{i=1}^n$ are "more likely" to be linearly separable (i.e. separable by an hyper-plan in $\mathbb{R}^m$) than the original observations $\{x_i^0\}_{i=1}^n$.

Based on this idea, a standard machine learning approach to tackle the classification task we are considering in this example is to choose a feature map $\Phi : \mathbb{R}^2 \to \mathbb{R}^m$, with $m \gg 1$, and to assume that

$$\mathbb{P}(Z = 1; x) \in \left\{ \frac{1}{1 + e^{-\alpha - \beta^\top \Phi(x)}},\ (\alpha, \beta) \in \mathbb{R} \times \mathbb{R}^m \right\}. \tag{4.12}$$

For this example we let Φ be the feature maps associated to the Gaussian kernel $k(x, x') = \exp(-\|x - x'\|^2/c)$, with $c > 0$ a tuning parameter, in which case $m = \infty$.

As we saw in Chapter 1 (pages 31-32), we can choose a $q \in \mathbb{N}$ and estimate (α, β) in (4.12) using principal components regression (with observations $\{\Phi(x_i^0)\}_{i=1}^n$) which, in term of predictions, is equivalent to using the model

$$\mathbb{P}(Z = 1; x) \in \left\{ \frac{1}{1 + e^{-\alpha - \gamma^\top \tilde{y}_n^{(q)}(x)}},\ (\alpha, \gamma) \in \mathbb{R} \times \mathbb{R}^q \right\}$$

where $\tilde{y}_n^{(q)}(x)$ is as defined in (4.8).

Using KPCA within logistic regression: results

The table below reports the percentage of correct predictions on the test set obtained for $q = q_{0.9}$ (with q_η as defined in (1.11)) and for different values of c.

c	$q_{0.9}$	Accuracy
0.1	77	76.00%
1	21	81.33%
6	5	84.33%
10	5	84.00%

From the above table we observe that

- Fitting a logistic regression model using the transformed observations $\{\Phi(x_i^0)\}_{i=1}^n$ leads to a meaningful classifier. More precisely, the mapping Φ successfully makes the observations $\{\Phi(x_i^0)\}_{i=1}^n$ "more linearly separable" than the original observations $\{x_i^0\}_{i=1}^n$.

- The value of q_η depends greatly on the choice of c, that is on the choice of the kernel. In other words, the dimension q of the linear subspace needed to represent well the infinite dimensional observations $\{\Phi(x_i^0)\}_{i=1}^n$ depends heavily on the choice of the kernel.

Digression: An other approach for estimating (4.12)

Recall the notation $\phi_i^0 = \Phi_i(x_i^0)$ and $\mathbf{\Phi}^0 = [\phi_{il}^0]$, and let

$$L((\alpha,\beta),\mathbf{\Phi}^0) = -\sum_{i=1}^{n} \log\left(1 + e^{-\alpha-\beta^\top \Phi(x_i^0)}\right)$$

be the log-likelihood function associated to the model (4.12).

Remark that

$$\mathbb{R}^m = \text{span}\{\phi_i^0\}_{i=1}^n \oplus \left(\text{span}\{\phi_i^0\}_{i=1}^n\right)^\perp$$

showing that for all $b \in \mathbb{R}^m$ there exist a $v_b \in \mathbb{R}^n$ and an $r_b \in \left(\text{span}\{\phi_i^0\}_{i=1}^n\right)^\perp$ such that $b = \left(\mathbf{\Phi}^0\right)^\top v_b + r_b$.

Remark: $\left(\text{span}\{\phi_i^0\}_{i=1}^n\right)^\perp \neq \emptyset$ when $m > n$.

Therefore, for all $i \in \{1,\dots,n\}$ we have

$$\beta^\top \phi_i^0 = \left((\mathbf{\Phi}^0)^\top v_\beta + r_\beta\right)^\top \phi_i^0 = \left((\mathbf{\Phi}^0)^\top v_\beta\right)^\top \phi_i^0$$

and thus, letting

$$\mathcal{B} = \left\{\beta \in \mathbb{R}^m : \exists v \in \mathbb{R}^n \text{ such that } \beta = \left(\mathbf{\Phi}^0\right)^\top v\right\},$$

it follows that

$$\text{argmax}_{(\alpha,\beta)\in\mathbb{R}\times\mathcal{B}} L((\alpha,\beta),\mathbf{\Phi}^0) \subseteq \text{argmax}_{\alpha\in\mathbb{R},\,\beta\in\mathbb{R}^m} L((\alpha,\beta),\mathbf{\Phi}^0). \quad (4.13)$$

Digression: An other approach for estimating (4.12)

To proceed further for $x \in \mathbb{R}^2$ we let $k_n(x) = (k(x_1, x), \dots, k(x_n, x))$.

Then, the above computations show that the problem of estimating (α, β) in (4.12) is equivalent to estimating (a, γ) in the model

$$\begin{aligned} &\mathbb{P}(Z = 1; x) \\ &\quad \in \left\{ \frac{1}{1 + e^{-a-\gamma^\top k_n(x)}},\ (a, \gamma) \in \mathbb{R} \times \mathbb{R}^n \right\}. \end{aligned} \tag{4.14}$$

Given an estimate $(\hat{a}, \hat{\gamma})$ in (4.14) the corresponding estimate of (α, β) in (4.12) is

$$(\hat{\alpha}, \hat{\beta}) := (\hat{a}, (\mathbf{\Phi}^0)^\top \hat{\gamma}).$$

Note also that, for $x \in \mathbb{R}^2$, we have

$$\hat{\beta}^\top \Phi^0(x) = \hat{\gamma}^\top k_n(x)$$

and, consequently, the estimate of $\mathbb{P}(Z = 1; x)$ obtained with the model (4.12) and the estimated parameter value $(\hat{\alpha}, \hat{\beta})$ is given by

$$\widehat{\mathbb{P}}(Z = 1; x) := (1 + e^{-\hat{a}-\hat{\gamma}^\top k_n(x)})$$

and thus does not require to explicitly compute $\hat{\beta}$.

Remark: In (4.14) there are as many observations as parameters as observations. To avoid overfitting we can estimate γ using principal component regression or penalized regression methods (see Chapter 6 and Chapter 7).

References

[1] Bishop, C. M. (2006). Pattern recognition. *Machine learning*, 128(9).

[2] Comon, P. (1994). Independent component analysis, a new concept? *Signal processing*, 36(3):287–314.

[3] Hyvärinen, A. and Oja, E. (2000). Independent component analysis: algorithms and applications. *Neural networks*, 13(4-5):411–430.

[4] Mardia, K., Kent, J., and Bibby, J. (1979). *Multivariate analysis.* Probability and mathematical statistics. Academic Press Inc.

[5] Paulsen, V. I. and Raghupathi, M. (2016). *An Introduction to the Theory of Reproducing Kernel Hilbert Spaces.* Cambridge Studies in Advanced Mathematics. Cambridge University Press.