Kernel Principal Component Analysis

Alex Beeny abeeny@siue.edu

Draft: March 23, 2024

Contents

Abstract

Principal component analysis (PCA) and kernel methods are tools often used in data science. The underlying theory of these tools depend on the properties of a special type of Hilbert space called a reproducing kernel Hilbert space (RKHS). This paper explores the essence of RKHSs using data science examples, in particular, PCA and kernel PCA. When kernel methods are applied to PCA, we can analyze nonlinear data in a high-dimensional feature space with some nice properties.

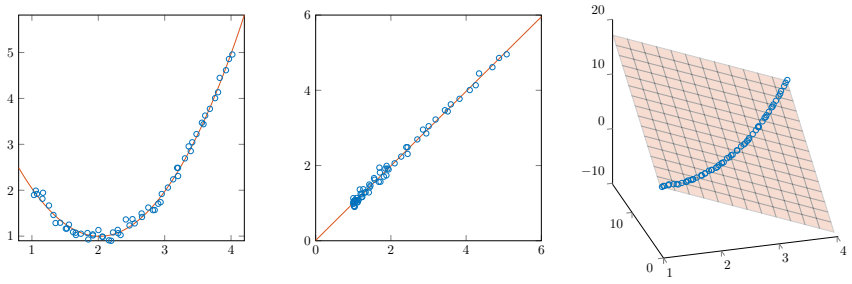

Figure 1: Plotting y against x (left) and the derived feature $z = \phi(x)$ (middle). The 3D plot (right) graphs the output y against x and x^2

1 Introduction

In linear regression, the equation of a line $y = a_0 + a_1 x$ is used to model observations based on training data. Here, the input variable x is used to predict the response variable y. The parameters a_0 and a_1 are chosen such that the residual error[1] is minimized. It seems natural to model data using polyomial equations in a similar way, that is, determine $a_0, a_1, \ldots, a_n$ such that

$$y = a_0 + a_1 x + a_2 x^2 + \cdots + a_n x^n \tag{1}$$

minimizes the residual error.

In multiple linear regression, the equation of a hyperplane

$$y = a_0 + a_1 x_1 + a_2 x_2 + \cdots + a_n x_n \tag{2}$$

is used to predict y using inputs $x_1, x_2, \ldots x_n$. It follows that these observations are points in $n + 1$ dimensions.

Example 1.1. Suppose we have a set of points $\left\{\left(x^{(i)}, y^{(i)}\right)\right\}_{i=1}^{k}$ in $\mathbb{R}^2$. Using polynomial regression, we fit the model $y \sim 1 + x + x^2$. We can derive a new variable from x to get

$$z = \phi(x) = a_0 + a_1 x + a_2 x^2.$$

By transforming our original data, the quadradic relationship between x and y can be viewed as a linear relationship between z and y. This shows that polynomial regression is just a special case of multiple regression where each power of x is treated as a separate dimension. See Figure 1

Here, we make the distinction between different kinds of input variables. We say that x is an **attribute** and that z is a **feature**.

[1] The residual for a given observation $\left(x^{(i)}, y^{(i)}\right)$ is $\left|y^{(i)} - \left(a_0 + a_1 x^{(i)}\right)\right|$.

2 Principal Component Analysis

Principal component analysis (PCA) is a coordinate transform that minimizes projection residuals along each of the major axes $v_1, v_2, \ldots, v_d$ of the transformed space. These axes are referred to as *principal components*. After the PCA transform is applied to a set of observations, the smallest projection error is along the first principal component. If we project down to two dimensions, the first two principal components have the smallest projection error. In general, projecting down to $p \leq d$ dimensions is most accurate along $v_1, v_2, \ldots, v_p$.

Conversely, projecting along $v_2, \ldots, v_d$ will result in the largest residual error in $d-1$ dimensions. It follows that v_1 must be the direction with the highest variance. Then v_2 is the direction with the next highest variance, and so on.

2.1 Minimizing projection residuals

Algorithm 1: Principal component analysis

Input: Data matrix $A \in \mathbb{R}^{n \times d}$ and number of components $p \leq d$
Output: Coefficient matrix $V_p \in \mathbb{R}^{d \times p}$ and transformed data $\tilde{A} \in \mathbb{R}^{n \times p}$

`// Center the data`
$A_0 \leftarrow A - \operatorname{col\,mean}(A)$;
`// Compute` $d \times d$ `covariance matrix`
$C \leftarrow \dfrac{1}{n-1} A_0^\top A$;
`// Diagonalize covariance matrix` $CV = VD$
$[V, D] \leftarrow \operatorname{eig}(C)$ where
$V = \begin{bmatrix} v_1, & v_2, & \ldots, & v_d \end{bmatrix}$ such that $v_1, v_2, \ldots, v_d \in \mathbb{R}^d$ and
$D = \operatorname{diag}(\lambda_1, \lambda_2, \ldots, \lambda_d)$ such that $\lambda_1 \geq \lambda_2 \geq \cdots \geq \lambda_d$;
`// Reduce dimension`
$V_p \leftarrow \begin{bmatrix} v_1, & v_2, & \ldots, & v_p \end{bmatrix}$;
`// Transform data`
$\tilde{A} \leftarrow A_0 V_p$;

Example 2.1. Consider the following matrix

$$A = \begin{bmatrix} 5 & 3 & 6 & 7 & 6 \\ 4 & 5 & 7 & 1 & 3 \\ 5 & 7 & 6 & 1 & 0 \\ 6 & 10 & 12 & 12 & 11 \\ 9 & 10 & 12 & 13 & 9 \end{bmatrix}.$$

The column means are $\mu = [5.8, 7, 8.6, 6.8, 5.8]$. Then the mean-centered data

becomes

$$X = A - \mu = \frac{1}{5}\begin{bmatrix} -4 & -20 & -13 & 1 & 1 \\ -9 & -10 & -8 & -29 & -14 \\ -4 & 0 & -13 & -29 & -29 \\ 1 & 15 & 17 & 26 & 26 \\ 16 & 15 & 17 & 31 & 16 \end{bmatrix}.$$

The covariance matrix is

$$C = X^T X = \frac{1}{5}\begin{bmatrix} 74 & 85 & 93 & 179 & 104 \\ 85 & 190 & 170 & 225 & 150 \\ 93 & 170 & 196 & 313 & 238 \\ 179 & 225 & 313 & 664 & 484 \\ 104 & 150 & 238 & 484 & 394 \end{bmatrix}.$$

Diagonalizing C gives

$$V = \begin{bmatrix} 0.1888 & -0.2020 & -0.6366 & 0.5495 & -0.4651 \\ 0.2755 & -0.7886 & 0.1472 & -0.4502 & -0.2791 \\ 0.3606 & -0.3464 & 0.3128 & 0.5836 & 0.5582 \\ 0.6979 & 0.2522 & -0.4422 & -0.3707 & 0.3411 \\ 0.5209 & 0.3922 & 0.5288 & 0.1316 & -0.5271 \end{bmatrix},$$

$$D = \begin{bmatrix} 264.8458 & 0 & 0 & 0 & 0 \\ 0 & 27.9766 & 0 & 0 & 0 \\ 0 & 0 & 9.3198 & 0 & 0 \\ 0 & 0 & 0 & 1.4579 & 0 \\ 0 & 0 & 0 & 0 & 0 \end{bmatrix}.$$

If we keep all 5 principal component vectors, then $V_5 = V$ and the projection of X along V is

$$P = XV = \begin{bmatrix} -1.9469 & 4.3453 & -0.8756 & -0.2039 & 0 \\ -6.9742 & -0.0660 & 1.4352 & 0.7590 & 0 \\ -8.1577 & -2.6752 & -0.8063 & -0.5704 & 0 \\ 8.4282 & -0.2330 & 1.8282 & -0.4996 & 0 \\ 8.6507 & -1.3711 & -1.5815 & 0.5149 & 0 \end{bmatrix}.$$

Here, the last column of P is the zero vector because the last eigenvalue of C is zero[2]. To perfectly reconstruct A, we need $k = 4$ principal components and the row vector μ

$$A = PV^T + \mu = PV_4^T + \mu.$$

If we use $k = 3$ principal components, then the projection of X onto V_3 is

$$P = XV_3 = \begin{bmatrix} -1.9469 & 4.3453 & -0.8756 \\ -6.9742 & -0.0660 & 1.4352 \\ -8.1577 & -2.6752 & -0.8063 \\ 8.4282 & -0.2330 & 1.8282 \\ 8.6507 & -1.3711 & -1.5815 \end{bmatrix}$$

[2]Since we subtracted the column means from a square matrix A, the dimension of the row space was reduced to 4.

and A is approximately reconstructed by

$$A \approx PV_3^T + \mu = \begin{bmatrix} 5.1 & 2.9 & 6.1 & 6.9 & 6.0 \\ 3.6 & 5.3 & 6.6 & 1.3 & 2.9 \\ 5.3 & 6.7 & 6.3 & 0.8 & 0.1 \\ 6.3 & 9.8 & 12.3 & 11.8 & 11.1 \\ 8.7 & 10.2 & 11.7 & 13.2 & 8.9 \end{bmatrix}.$$

We can compute the reconstruction error using

$$E_k = \|A - (PV_k^T + \mu)\|_F,$$

where $\|\cdot\|_F$ is the Frobenius norm. By the SVD, we have $X = USV^T$, where $S = \sqrt{D}$. So, the projection of X onto V_k is

$$P = XV_k = US_k,$$

where S_k is the diagonal matrix of the first k singular values. Then the reconstruction error becomes

$$\begin{aligned} \|A - (PV_k^T + \mu)\|_F &= \|(A - \mu) - PV_k^T\|_F \\ &= \|X - PV_k^T\|_F \\ &= \|USV^T - US_kV^T\|_F \\ &= \|U(S - S_k)V^T\|_F \\ &= \|S - S_k\|_F \\ &= \sigma_k + \sigma_{k+1} + \cdots + \sigma_p. \end{aligned}$$

Hence,

$$E_3 = \sigma_3 + \sigma_4 = \sqrt{1.4579} + 0 = 1.2074.$$

2.2 Linear regression and PCA

[15] Let $\mathbf{x}_1, \ldots, \mathbf{x}_m \in \mathbb{R}^n$ be observation vectors and $\mathbf{y} \in \mathbb{R}^n$ be a target vector. A linear regression model finds a vector of weights $\mathbf{w} = (w_1, w_2, \ldots, w_n)$ to determine the linear function

$$f(\mathbf{x}) = \mathbf{w} \cdot \mathbf{x} = \sum_{i=1}^{n} w_i x_i, \tag{3}$$

where $\mathbf{x} = (x_1, x_2, \ldots, x_n)$ is an input vector. The residual error of each observation is $\mathbf{y} - f(\mathbf{x}_i)$, for $i = 1, 2, \ldots, m$. Then the residual sum of squares is given to be

$$RSS = \sum_{i=1}^{m} (\mathbf{y} - f(\mathbf{x}_i))^2 = \sum_{i=1}^{m} (\mathbf{y} - \mathbf{w} \cdot \mathbf{x}_i)^2 = (\mathbf{y} - X^\top \mathbf{w})^\top (\mathbf{y} - X^\top \mathbf{w}), \tag{4}$$

where $X = [\mathbf{x}_1, \ldots, \mathbf{x}_m]$. By minimizing RSS, the magnitude of the residuals will be as small as possible, producing the optimal linear model $f(\mathbf{x})$. Therefore, this method is known as least squares regression. Differentiate (4) with respect to $\mathbf{w}$ and set equal to zero so that

$$2X^\top \mathbf{y} - 2X^\top X\mathbf{w} = 0. \tag{5}$$

This leads to the normal equation $X^\top X\mathbf{w} = X^\top \mathbf{y}$. Thus,

$$\mathbf{w} = (X^\top X)^{-1} X^\top \mathbf{y} \tag{6}$$

gives the optimal linear model which minimizes residual error.

Example 2.2. In two dimensions, let $\mathbf{x} = (x_1, \ldots, x_m)$ and $\mathbf{y} = (y_1, \ldots, y_m)$. Then the least squares regression model is given by

$$f(x) = \frac{\operatorname{cov}(\mathbf{x}, \mathbf{y})}{\operatorname{var}(\mathbf{x})}(x - \overline{\mathbf{x}}) + \overline{\mathbf{y}}, \tag{7}$$

Since $\mathbf{x}$ is treated as an input variable and $\mathbf{y}$ is treated as an output, this is a type of supervised learning. In contrast, PCA is an unsupervised learning technique. PCA organizes the variables in the input space to reveal any patterns in the underlying data. If we have input variables $\mathbf{x}_1$ and $\mathbf{x}_2$, we can use PCA to find a linear pattern among the inputs without trying to predict an output. First, we find the direction of the largest variance $\lambda_{\max}$ using the covariance matrix

$$\begin{bmatrix} \operatorname{cov}(\mathbf{x}_1, \mathbf{x}_1) & \operatorname{cov}(\mathbf{x}_1, \mathbf{x}_2) \\ \operatorname{cov}(\mathbf{x}_2, \mathbf{x}_1) & \operatorname{cov}(\mathbf{x}_2, \mathbf{x}_2) \end{bmatrix} = \begin{bmatrix} \operatorname{var}(\mathbf{x}_1) & \operatorname{cov}(\mathbf{x}_1, \mathbf{x}_2) \\ \operatorname{cov}(\mathbf{x}_1, \mathbf{x}_2) & \operatorname{var}(\mathbf{x}_2) \end{bmatrix}.$$

In particular,

$$\lambda_{\max} = \frac{1}{2}\left(\operatorname{var}(\mathbf{x}) + \operatorname{var}(\mathbf{y}) + \sqrt{(\operatorname{var}(\mathbf{x}) - \operatorname{var}(\mathbf{y}))^2 + 4\operatorname{cov}(\mathbf{x}, \mathbf{y})^2}\right).$$

Then the PCA regression model becomes

$$g(x) = \frac{\lambda_{\max} - \operatorname{var}(\mathbf{x}_1)}{\operatorname{cov}(\mathbf{x}_1, \mathbf{x}_2)}(x - \overline{\mathbf{x}}_1) + \overline{\mathbf{x}}_2. \tag{8}$$

In this case, the projection residuals are orthogonal to the PCA regression line. See Figure 2.

3 Reproducing Kernel Hilbert Space

In this section, our goal is to establish properties of Hilbert spaces and kernel functions that can be used to modify the PCA algorithm. To begin, we will briefly cite some definitions and results from analysis [8], [11] and matrix theory [6].

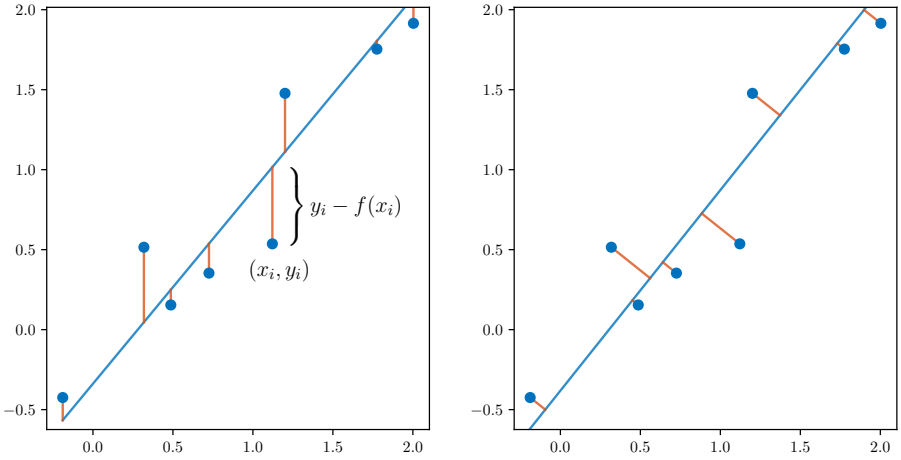

Figure 2: Least squares regression model $f(x)$ (left) minimizes the sum of squared errors while total least squares, i.e., the PCA model $g(x)$ (right) minimizes the orthogonal projections.

Definition 3.1. [16] Let X be a (real) vector space. An *inner product* is a function $\langle \cdot, \cdot \rangle : X \times X \to \mathbb{R}$ which satisfies the following properties:

1. Symmetry. For all $x, y \in X$,
$$\langle x, y \rangle = \langle y, x \rangle.$$
2. Linear in the first argument. For all $x, y, z \in X$, $\alpha, \beta \in \mathbb{R}$,
$$\langle \alpha x + \beta y, z \rangle = \alpha \langle x, z \rangle + \beta \langle y, z \rangle.$$
3. Positive definite. For all $x \in X$,
$$\langle x, x \rangle \geq 0$$
and $\langle x, x \rangle = 0$ if and only if $x = 0$.

An *inner product space* is a vector space along with an inner product.

Since real inner products are symmetric and linear in the first argument,

$$\langle z, \alpha x + \beta y \rangle = \langle \alpha x + \beta y, z \rangle = \alpha \langle x, z \rangle + \beta \langle y, z \rangle = \alpha \langle z, x \rangle + \beta \langle z, y \rangle.$$

So, we also have linearity in the second argument.

The norm induced by an inner product is defined as

$$\|x\| = \langle x, x \rangle^{1/2}$$

and the metric induced by this norm is

$$d(x, y) = \|x - y\| = \langle x - y, x - y\rangle^{1/2}.$$

It follows that an inner product space is also a normed space and a metric space. So, the induced norm will have the following properties for all $x, y \in X$ and $\alpha \in \mathbb{R}$:

1. Triangle inequality. $\|x + y\| \leq \|x\| + \|y\|$;
2. Scalar multiplication. $\|\alpha x\| = |\alpha|\|x\|$;
3. Positivity. $\|x\| \geq 0$ and $\|x\| = 0$ if and only if $x = 0$.

Definition 3.2 (Hilbert space)**.** A Hilbert space is a complete inner product space. For a Hilbert space H, we sometimes denote the inner product as $\langle \cdot, \cdot \rangle_H$ to avoid ambiguity.

Two Hilbert spaces H and L (over the same field) are said to be *isomorphic* if there is a bijection $T : H \to L$ such that

$$\langle x, y\rangle_H = \langle Tx, Ty\rangle_L, \tag{9}$$

for every $x, y \in H$. In [8], Kreyszig shows that two Hilbert spaces are isomorphic if and only if they have the same dimension. If V is an inner product space that is not complete, then it can be extended to a Hilbert space by completion. The completion of an inner product space is denoted as $\overline{V}$ and is unique up to isomorphism.

A Hilbert space is said to be *separable* if it contains a dense countable subset. It can be shown [8] that a Hilbert space is separable if and only if it has a countable orthonormal basis. The following example demonstrates a useful property of separable Hilbert spaces.

Example 3.3. [11] The space of square-summable (real) sequences is defined as

$$\ell^2(A) = \left\{ x : A \to \mathbb{R} \;\middle|\; \sum_{a \in A} x_a^2 < \infty \right\}. \tag{10}$$

Given the inner product

$$\langle x, y\rangle = \sum_{a \in A} x_a y_a, \tag{11}$$

$\ell^2(A)$ is a Hilbert space. Moreover, $\ell^2(A)$ is separable if and only if A is countable. It follows that the sequence space $\ell^2 = \ell^2(\mathbb{N})$ is the separable Hilbert space of square-summable sequences. Due to the Riesz-Fischer theorem, every infinite-dimensional Hilbert space is isomorphic to ℓ^2.

Definition 3.4 (Gram matrix)**.** [6] Let $x_1, x_2, \ldots, x_n$ be vectors in an inner product space X. The matrix $G \in \mathbb{R}^{n \times n}$ is called a *Gram matrix* if $[G]_{ij} = \langle x_i, x_j\rangle$.

Example 3.5. Consider the vectors in $\mathbb{R}^3$:

$$\mathbf{v}_1 = \begin{bmatrix} v_{11} \\ v_{21} \\ v_{31} \end{bmatrix}, \qquad \mathbf{v}_2 = \begin{bmatrix} v_{12} \\ v_{22} \\ v_{32} \end{bmatrix}, \qquad \mathbf{v}_3 = \begin{bmatrix} v_{13} \\ v_{23} \\ v_{33} \end{bmatrix}, \qquad \mathbf{v}_4 = \begin{bmatrix} v_{14} \\ v_{24} \\ v_{34} \end{bmatrix}.$$

The Gram matrix for these vectors is

$$G = \begin{bmatrix} \mathbf{v}_1^\top \mathbf{v}_1 & \mathbf{v}_1^\top \mathbf{v}_2 & \mathbf{v}_1^\top \mathbf{v}_3 & \mathbf{v}_1^\top \mathbf{v}_4 \\ \mathbf{v}_2^\top \mathbf{v}_1 & \mathbf{v}_2^\top \mathbf{v}_2 & \mathbf{v}_2^\top \mathbf{v}_3 & \mathbf{v}_2^\top \mathbf{v}_4 \\ \mathbf{v}_3^\top \mathbf{v}_1 & \mathbf{v}_3^\top \mathbf{v}_2 & \mathbf{v}_3^\top \mathbf{v}_3 & \mathbf{v}_3^\top \mathbf{v}_4 \\ \mathbf{v}_4^\top \mathbf{v}_1 & \mathbf{v}_4^\top \mathbf{v}_2 & \mathbf{v}_4^\top \mathbf{v}_3 & \mathbf{v}_4^\top \mathbf{v}_4 \end{bmatrix}.$$

If V is a matrix whose columns are $\mathbf{v}_1$, $\mathbf{v}_2$, $\mathbf{v}_3$, $\mathbf{v}_4$, then we can write $G = V^\top V$.

Definition 3.6 (kernel)**.** Let X be a nonempty set and $k : X \times X \to \mathbb{R}$. Then k is a (positive semi-definite) *kernel* if

1. k is symmetric: for all $x, y \in X$,

$$k(x, y) = k(y, x);$$

2. k is positive semi-definite: if $x_1, \ldots, x_n \in X$ and $c_1, \ldots, c_n \in \mathbb{R}$, then

$$\sum_{i=1}^{n} \sum_{j=1}^{n} c_i c_j k(x_i, x_j) \geq 0.$$

Equivalently, the matrix $K \in \mathbb{R}^{n \times n}$ whose entries are $[K]_{ij} = k(x_i, x_j)$ is positive semi-definite, that is, $\mathbf{c}^\top K \mathbf{c} \geq 0$ for all $\mathbf{c} \in \mathbb{R}^n$. We call K a *kernel matrix*. Note that since k is symmetric, so is K.

Definition 3.7 (feature map)**.** [10] Let F be a Hilbert space of functions $f : X \to \mathbb{R}$. A feature map is a function $\Phi : X \to F$.

Our goal is to have kernels written as inner products of feature maps:

$$k(x, y) = \langle \Phi(x), \Phi(y) \rangle_{H_k}.$$

If Φ is a known feature map, then k would surely be symmetric and positive definite since inner products are. If instead we know k is a kernel, then we need to show that there is a unique Hilbert space with the desired inner product.

3.1 Constructing a reproducing kernel Hilbert space

3.2 Constructing kernels

Theorem 3.8. *[10, 15] Suppose k_1 and k_2 are kernels over $X \times X$. The following functions kernels.*

1. $k(x,y) = a_1 k_1(x,y) + a_2 k_2(x,y)$ *for all* $a_1, a_2 \geq 0$.
2. $k(x,y) = k_1(x,y)k_2(x,y)$.
3. $k(x,y) = a_0 + a_1 k_1(x,y) + a_2 k_1(x,y)^2 + \cdots + a_n k_1(x,y)^n$ *for all* $n \in \mathbb{N}$ *and* $a_0, \ldots, a_n \geq 0$.
4. $k(x,y) = k_1(h(x), h(y))$ *for all* $h : X \to X$.
5. $k(x,y) = g(x)g(y)$ *for all* $g : X \to \mathbb{R}$.
6. $k(x,y) = \exp(k_1(x,y))$.

Proof. Let $x_1, \ldots, x_n \in X$ and $c_1, \ldots, c_n \in \mathbb{R}$.

1. Let $k = a_1 k_1 + a_2 k_2$ for $a_1, a_2 \geq 0$. Since k_1 and k_2 are symmetric,

   $$k(x,y) = a_1 k_1(x,y) + a_2 k_2(x,y) = a_1 k_1(y,x) + a_2 k_2(y,x) = k(y,x),$$

   for all $x, y \in X$. So, k is symmetric.

   Since k_1 and k_2 are positive semi-definite and $a_1, a_2 \geq 0$,

   $$\begin{aligned}\sum_{i=1}^n \sum_{j=1}^n c_i c_j k(x_i, x_j) &= \sum_{i=1}^n \sum_{j=1}^n c_i c_j (a_1 k_1(x_i, x_j) + a_2 k_2(x_i, x_j)) \\ &= a_1 \sum_{i=1}^n \sum_{j=1}^n c_i c_j k_1(x_i, x_j) + a_2 \sum_{i=1}^n \sum_{j=1}^n c_i c_j k_2(x_i, x_j) \\ &\geq 0.\end{aligned}$$

   So, k is positive semi-definite.

2. Let $k = k_1 k_2$. Define K so that $[K]_{ij} = k(x_i, x_j) = k_1(x_i, x_j)k_2(x_i, x_j)$. Let K_1 and K_2 be the Gram matrices for k_1 and k_2, respectively. Then K_1, K_2 have orthonormal eigenvectors and nonnegative eigenvalues such that

   $$\begin{aligned}K_1 &= VLV^\top \\ &= \begin{bmatrix} v_{11} & \cdots & v_{1n} \\ \vdots & \ddots & \vdots \\ v_{n1} & \cdots & v_{nn} \end{bmatrix} \begin{bmatrix} \lambda_1 & \cdots & 0 \\ \vdots & \ddots & \vdots \\ 0 & \cdots & \lambda_n \end{bmatrix} \begin{bmatrix} v_{11} & \cdots & v_{n1} \\ \vdots & \ddots & \vdots \\ v_{1n} & \cdots & v_{nn} \end{bmatrix} \\ &= \begin{bmatrix} \sum_{j=1}^n \lambda_j v_{1j} v_{1j} & \cdots & \sum_{j=1}^n \lambda_j v_{nj} v_{1j} \\ \vdots & \ddots & \vdots \\ \sum_{j=1}^n \lambda_j v_{1j} v_{nj} & \cdots & \sum_{j=1}^n \lambda_j v_{nj} v_{nj} \end{bmatrix} \\ &= \sum_{j=1}^n \lambda_j \begin{bmatrix} v_{1j} v_{1j} & \cdots & v_{nj} v_{1j} \\ \vdots & \ddots & \vdots \\ v_{1j} v_{nj} & \cdots & v_{nj} v_{nj} \end{bmatrix}\end{aligned}$$

and

$$K_2 = UMU^\top = \sum_{j=1}^{n} \mu_j \begin{bmatrix} u_{1j}u_{1j} & \cdots & u_{nj}u_{1j} \\ \vdots & \ddots & \vdots \\ u_{1j}u_{nj} & \cdots & u_{nj}u_{nj} \end{bmatrix}.$$

Let $\mathbf{v}_i = \begin{bmatrix} v_{1i} & \cdots & v_{ni} \end{bmatrix}^\top$ and $\mathbf{u}_j = \begin{bmatrix} u_{1j} & \cdots & u_{nj} \end{bmatrix}$, for all $i, j = 1, 2, \ldots, n$. Then

$$\begin{aligned}
K &= K_1 \circ K_2 \\
&= \sum_{i=1}^{n} \lambda_i \begin{bmatrix} v_{1i}v_{1i} & \cdots & v_{ni}v_{1i} \\ \vdots & \ddots & \vdots \\ v_{1i}v_{ni} & \cdots & v_{ni}v_{ni} \end{bmatrix} \circ \sum_{j=1}^{n} \mu_j \begin{bmatrix} u_{1j}u_{1j} & \cdots & u_{nj}u_{1j} \\ \vdots & \ddots & \vdots \\ u_{1j}u_{nj} & \cdots & u_{nj}u_{nj} \end{bmatrix} \\
&= \sum_{i=1}^{n}\sum_{j=1}^{n} \lambda_i\mu_j \begin{bmatrix} v_{1i}u_{1j}v_{1i}u_{1j} & \cdots & v_{1i}u_{1j}v_{ni}u_{nj} \\ \vdots & \ddots & \vdots \\ v_{ni}u_{nj}v_{1i}u_{1j} & \cdots & v_{ni}u_{nj}v_{ni}u_{nj} \end{bmatrix} \\
&= \sum_{i=1}^{n}\sum_{j=1}^{n} \lambda_i\mu_j \begin{bmatrix} v_{1i}u_{1j} \\ \vdots \\ v_{ni}u_{nj} \end{bmatrix} \begin{bmatrix} v_{1i}u_{1j} & \cdots & v_{ni}u_{nj} \end{bmatrix} \\
&= \sum_{i=1}^{n}\sum_{j=1}^{n} \lambda_i\mu_j (\mathbf{v}_i \circ \mathbf{u}_j)(\mathbf{v}_i \circ \mathbf{u}_j)^\top,
\end{aligned}$$

where $\circ$ is the Hadamard product. Each $(\mathbf{v}_i \circ \mathbf{u}_j)(\mathbf{v}_i \circ \mathbf{u}_j)^\top$ is a symmetric positive semi-definite matrix. Since K_1, K_2 are positive semi-definite, we have $\lambda_i, \mu_i > 0$. Then K is symmetric positive semi-definite.

3. By part 2, $k_1, k_1^2, \ldots, k_1^n$ are kernels. By part 1, $a_0 + a_1k_1 + a_2k_1^2 + \cdots + a_nk_1^n$ is a kernel.

4. Since $y_i = h(x_i) \in X$ for all $i = 1, 2, \ldots, n$, we have

$$\begin{aligned}
\sum_{i=1}^{n}\sum_{j=1}^{n} c_ic_jk(x_i, x_j) &= \sum_{i=1}^{n}\sum_{j=1}^{n} c_ic_jk_1(h(x_i), h(x_j)) \\
&= \sum_{i=1}^{n}\sum_{j=1}^{n} c_ic_jk_1(y_i, y_j) \\
&\geq 0.
\end{aligned}$$

5. Let $g : X \to \mathbb{R}$ and let $c_i g(x_i) = y_i \in \mathbb{R}$. If $k(x, y) = g(x)g(y)$, then

$$\begin{aligned}\sum_{i=1}^{n}\sum_{j=1}^{n} c_i c_j k(x_i, x_j) &= \sum_{i=1}^{n}\sum_{j=1}^{n} c_i g(x_i) c_j g(x_j) \\ &= \sum_{i=1}^{n}\sum_{j=1}^{n} y_i y_j \\ &= \left(\sum_{i=1}^{n} y_i\right)^2 \\ &\geq 0.\end{aligned}$$

6. Let K_1 be the Gram matrix for k_1. If $K_1 v = \lambda v$, then $K_1^m = \lambda^m v$ for all $m \in \mathbb{N}$. So,

$$(\exp K_1)v = \sum_{m=0}^{\infty} \frac{K_1^m v}{m!} = \sum_{m=0}^{\infty} \frac{\lambda^m v}{m!} = e^{\lambda} v.$$

   Then $K = \exp K_1$ has eigenvalues e^{λ}. Since K_1 is positive semi-definite, it has real eigenvalues so that $e^{\lambda} > 0$. It follows that K is positive definite.

□

Theorem 3.9 (Gaussian kernel)**.** *The function $k : \mathbb{R}^n \times \mathbb{R}^n \to \mathbb{R}$ defined by*

$$k(x, y) = \exp\left(\frac{-\|x - y\|_2^2}{\sigma^2}\right),$$

is a kernel.

Proof. □

4 Kernel PCA

Recall that linear PCA finds new components that reveal more information about the structure of high-dimensional data. Since PCA is an orthogonal projection, the original data is rotated witin the original space of input variables. The work of Schölkopf, Smola, and Müller [12, 13] generalized PCA based on the successful application of kernel methods in support vector machines. In kernel PCA, the inner product of the input space is replaced with the inner product of a feature space. As such, the principal components of kernel PCA are nonlinear transformations of input variables, or features.

Using results from the previous section, a kernel function k defines a unique reproducing kernel Hilbert space H_k and feature map $\Phi : X \to H_k$ such that

$$k(x, y) = \langle \Phi(x), \Phi(y) \rangle, \tag{12}$$

for all $x, y \in H_k$.

4.1 Covariance matrix and kernel matrix

In PCA, we diagonalize $X^\top X$.

4.2 Centering in the feature space

Let $\Phi : X \to H_k$ be a feature map determined by a kernel k. Since Φ may be nonlinear, the image $\Phi(x)$ of a centered vector $x \in X$ is not guaranteed to be centered. For an effective PCA algorithm, it is necessary to compute the kernel matrix of centered vectors in the feature space. [13]

Given $x_1, \ldots, x_n \in X$, the points

$$\Phi_0(x_i) = \Phi(x_i) - \frac{1}{n}\sum_{i=1}^{n}\Phi(x_i), \quad \text{for } i = 1, \ldots, m \tag{13}$$

are the centered feature vectors in H_k. Then the centered kernel matrix becomes

$$\begin{aligned}
[K_0]_{ij} &= \langle \Phi_0(x_i), \Phi_0(x_j) \rangle \\
&= \left\langle \Phi(x_i) - \frac{1}{n}\sum_{p=1}^{n}\Phi(x_p), \Phi(x_j) - \frac{1}{n}\sum_{q=1}^{n}\Phi(x_q) \right\rangle \\
&= \langle \Phi(x_i), \Phi(x_j) \rangle - \frac{1}{n}\sum_{p=1}^{n}\langle \Phi(x_p), \Phi(x_j) \rangle \\
&\qquad - \frac{1}{n}\sum_{q=1}^{n}\langle \Phi(x_i), \Phi(x_q) \rangle \\
&\qquad + \frac{1}{n^2}\sum_{p=1}^{n}\sum_{q=1}^{n}\langle \Phi(x_p), \Phi(x_q) \rangle \\
&= [K]_{ij} - \frac{1}{n}\sum_{p=1}^{n}[K]_{pj} - \frac{1}{n}\sum_{q=1}^{n}[K]_{iq} + \frac{1}{n^2}\sum_{p=1}^{n}\sum_{q=1}^{n}[K]_{pq},
\end{aligned}$$

where K is the uncentered kernel matrix given by $[K]_{ij} = \langle \Phi(x_i), \Phi(x_j) \rangle$. Then the formula for the centered kernel matrix can be written as

$$K_0 = K - \operatorname{col\,mean}(K) - \operatorname{row\,mean}(K) + \operatorname{mean}(K). \tag{14}$$

See notes in Appendices A.1 and A.2.

Algorithm 2: Kernel PCA
Input: Data matrix $A \in \mathbb{R}^{n \times d}$, kernel function k, number of components $p \leq n$
Output: Transformed data $\tilde{A} \in \mathbb{R}^{n \times p}$

Example 4.1. Consider the problem of classifying points based on their radii. These points cannot be separated using a linear classifier in the two dimensions. However, by mapping them to a three-dimensional space, they can be separated by planes. Applying kernel PCA, these points can be sent to the RKHS associated with a Gaussian kernel without using an explicit feature map. The points in this high-dimensional feature space can then be projected onto the first three principal components to find separation boundaries. See Figure 3.

5 Conclusion

A Linear Algebra

A number of matrix definitions and results are presented without proof. Unless otherwise specified, let A be an $n \times d$ matrix over the real numbers.

1. A is *normal* if it is square and $AA^\top = A^\top A$.
2. Symmetric matrices are normal.
3. Symmetric matrices have orthogonal eigenvectors and real eigenvectors.
4. Positive semi-definite matrices have nonnegative eigenvalues.
5. Positive definite matrices have positive eigenvalues.
6. $A^\top A$ and $AA^\top$ are symmetric positive semi-definite.

A.1 Matrix operations and notation

Let A be an $n \times d$ matrix. We write $[A]_{ij}$ to indicate the matrix entry in the i-th row and the j-th column.

Definition A.1. Define the *entry-wise mean* of A as

$$\operatorname{mean}(A) = \frac{1}{nd} \sum_{i=1}^{n} \sum_{j=1}^{d} [A]_{ij}. \tag{15}$$

Define the *column-wise mean* of A as a $1 \times d$ row vector whose j-th entry is the mean of column j given by the formula

$$\begin{aligned} \operatorname{col\,mean}(A) &= \left[\tfrac{1}{n} \textstyle\sum_{i=1}^{n} [A]_{i1}, \quad \tfrac{1}{n} \sum_{i=1}^{n} [A]_{i2}, \quad \ldots, \quad \tfrac{1}{n} \sum_{i=1}^{n} [A]_{id} \right] \\ &= \frac{1}{n} \sum_{i=1}^{n} \left[[A]_{i1}, \quad [A]_{i2}, \quad \ldots, \quad [A]_{id} \right]. \end{aligned} \tag{16}$$

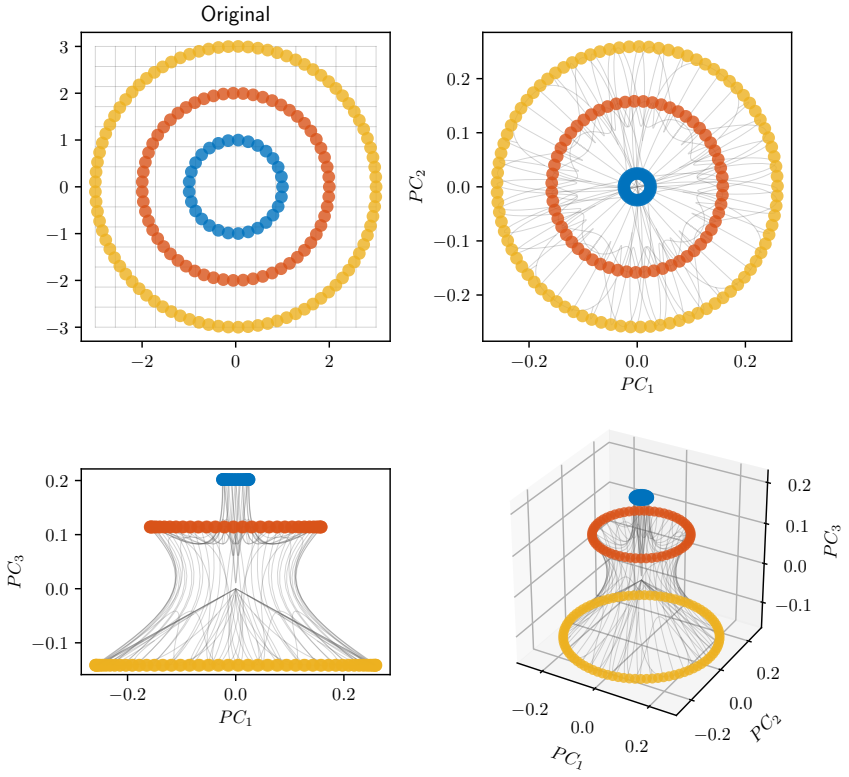

Figure 3: An idealized set of points in the plane are classified based on their radius. Kernel PCA with the Gaussian kernel is applied to find separation boundaries using the first three principal components.

Define the *row-wise mean* of A as an $n \times 1$ column vector whose i-th entry is the mean of row i given by the formula

$$\text{row mean}(A) = \begin{bmatrix} \frac{1}{d}\sum_{j=1}^{d}[A]_{1j} \\ \frac{1}{d}\sum_{j=1}^{d}[A]_{2j} \\ \vdots \\ \frac{1}{d}\sum_{j=1}^{d}[A]_{nj} \end{bmatrix} = \frac{1}{d}\sum_{j=1}^{d}\begin{bmatrix} [A]_{1j} \\ [A]_{2j} \\ \vdots \\ [A]_{nj} \end{bmatrix}. \tag{17}$$

Let $[a]_{p\times q}$ denote the $p \times q$ repeated matrix whose entries are all a. Then Equations (15) to (17) can be written as

$$\text{mean}(A) = \left[\tfrac{1}{n}\right]_{1\times n} \cdot A \cdot \left[\tfrac{1}{d}\right]_{d\times 1} \tag{18}$$

$$\text{col mean}(A) = \left[\tfrac{1}{n}\right]_{1\times d} \cdot A \tag{19}$$

$$\text{row mean}(A) = A \cdot \left[\tfrac{1}{d}\right]_{d\times 1}. \tag{20}$$

Definition A.2. Let A be an $n \times d$ matrix whose columns $\mathbf{a}_1, \mathbf{a}_2, \ldots, \mathbf{a}_d \in \mathbb{R}^n$ represent variables and rows represent observations. Then the *covariance matrix* of A is given by

$$\text{cov}(A) = \begin{bmatrix} \text{cov}(\mathbf{a}_1, \mathbf{a}_1) & \text{cov}(\mathbf{a}_1, \mathbf{a}_2) & \cdots & \text{cov}(\mathbf{a}_1, \mathbf{a}_d) \\ \text{cov}(\mathbf{a}_2, \mathbf{a}_1) & \text{cov}(\mathbf{a}_2, \mathbf{a}_2) & \cdots & \text{cov}(\mathbf{a}_2, \mathbf{a}_d) \\ \vdots & \vdots & \ddots & \vdots \\ \text{cov}(\mathbf{a}_d, \mathbf{a}_1) & \text{cov}(\mathbf{a}_d, \mathbf{a}_2) & \cdots & \text{cov}(\mathbf{a}_d, \mathbf{a}_d) \end{bmatrix}. \tag{21}$$

If A is centered, i.e., the columns all have mean zero, then we can write

$$\text{cov}(A) = \frac{1}{n-1}A^\top A. \tag{22}$$

A.2 Broadcasting

Consider the sum of two real matrices $A + B$. By definition, A and B must both have size $n \times d$. This means we cannot add a 2×3 matrix A and a 2×1 vector $\mathbf{b}$. However, in many programming languages the sum $A + \mathbf{b}$ would be handled using *broadcasting* [4]. In this case, $\mathbf{b}$ is converted to a 2×3 matrix $\begin{bmatrix}\mathbf{b} & \mathbf{b} & \mathbf{b}\end{bmatrix}$ so that normal matrix addition applies. Generally, broadcasting a vector $\mathbf{b} \in \mathbb{R}^n$ to an $n \times d$ matrix can be represented as the matrix product

$$\mathbf{b} \cdot [1]_{1\times d} = \begin{bmatrix} b_1 \\ b_2 \\ \vdots \\ b_n \end{bmatrix} \cdot \begin{bmatrix} 1 & 1 & \cdots & 1 \end{bmatrix} = \begin{bmatrix} b_1 & b_1 & \cdots & b_1 \\ b_2 & b_2 & \cdots & b_2 \\ \vdots & \vdots & & \vdots \\ b_n & b_n & \cdots & b_n \end{bmatrix}, \tag{23}$$

where the notation $[a]_{n\times d}$ represents an $n \times d$ matrix whose entries are all a.

Definition A.3. For an $n \times d$ matrix A, we can define addition by an $n \times 1$ column vector $\mathbf{c}$ as

$$A + \mathbf{c} := A + \mathbf{c} \cdot [1]_{1 \times d}. \tag{24}$$

Similarly, addition by a $1 \times d$ row vector $\mathbf{r}$ can be defined as

$$A + \mathbf{r} := A + [1]_{n \times 1} \cdot \mathbf{r} \tag{25}$$

and addition by a scalar a can be defined as

$$A + a := A + a \cdot [1]_{n \times d}. \tag{26}$$

The left hand sides of Equations (24) to (26) are more concise and intuitive than the right hand sides. Provided that the vector types are clearly defined and compatible, there should be no ambiguity when adding column vectors, row vectors, and scalars to matrices. Moreover, this method of broadcasting is consistent with scientific programming languages.

B Riesz Representation Theorem

Theorem B.1 (Riesz Representation Theorem)**.** *[16] Let $\phi : H \to \mathbb{R}$ be a continuous linear functional defined on a Hilbert space H. Then there exists a unique element $g \in H$ such that $\phi(g) = \langle f, g \rangle_H$ for all $g \in H$.*

Proof. □

C Mercer's Theorem

D Code

References

[1] Peter Bartlett. CS 281B / Stat 241B Statistical Learning Theory. Lecture notes, Spring 2008.

[2] Peter Bartlett. CS 281B / Stat 241B Statistical Learning Theory. Lecture notes, Spring 2014.

[3] Ji Chen and Xiaodong Li. Model-free nonconvex matrix completion: Local minima analysis and applications in memory-efficient kernel pca. *Journal of Machine Learning Research*, 20(142):1–39, 2019.

[4] Charles R Harris, K Jarrod Millman, Stéfan J Van Der Walt, Ralf Gommers, Pauli Virtanen, David Cournapeau, Eric Wieser, Julian Taylor, Sebastian Berg, Nathaniel J Smith, et al. Array programming with numpy. *Nature*, 585(7825):357–362, 2020.

[5] Thomas Hofmann, Bernhard Schölkopf, and Alexander J. Smola. Kernel methods in machine learning. *The Annals of Statistics*, 36(3), jun 2008.

[6] R.A. Horn and C.R. Johnson. *Matrix Analysis*. Cambridge University Press, 2013.

[7] The MathWorks Inc. Statistics and machine learning toolbox, 2022.

[8] E. Kreyszig. *Introductory Functional Analysis with Applications*. Wiley Classics Library. Wiley, 1991.

[9] Mehryar Mohri, Afshin Rostamizadeh, and Ameet Talwalkar. *Foundations of Machine Learning*. Adaptive Computation and Machine Learning series. MIT Press, 2012.

[10] Cynthia Rudin. Intuition for the Algorithms of Machine Learning. Self-pub, 2020.

[11] Walter Rudin. *Real and Complex Analysis*. Higher Mathematics Series. McGraw-Hill Education, 1987.

[12] Bernhard Schölkopf, Alexander Smola, and Klaus-Robert Müller. Kernel principal component analysis. In *International conference on artificial neural networks*, pages 583–588. Springer, 1997.

[13] Bernhard Schölkopf, Alexander Smola, and Klaus-Robert Müller. Nonlinear component analysis as a kernel eigenvalue problem. *Neural computation*, 10(5):1299–1319, 1998.

[14] Cosma Rohilla Shalizi. Advanced Data Analysis from an Elementary Point of View. Draft textbook, 2021.

[15] John Shawe-Taylor and Nello Cristianini. *Kernel Methods for Pattern Analysis*. Cambridge University Press, 2004.

[16] Christopher G. Small and D.L. Mcleish. *Hilbert Space Methods in Probability and Statistical Inference*. John Wiley & Sons, Inc, 1994.