\[
\begin{align*}
\phi: \text{non-linear function} \\
\mathbf{K} = \phi(x)^T \phi(y) \\
K(x_i, x_j) = \phi(x_i)^T \phi(x_j)
\end{align*}
\]

If \( K \) matrix is positive semi-definite

Similarity or the kernel value between two points in input space corresponds to a dot product in some feature space

1. **Polynomial kernel** \((c, q)\)

\[
k(x, y) = (c + x^T y)^q
\]

- \( q = 1, c = 0 \) \( \Rightarrow \) **linear kernel**
- \( q = 3 \)
- \( c = 0 \) \( \Rightarrow \) homogeneous
- \( c \neq 0 \) \( \Rightarrow \) inhomogeneous
Input

PCA

=⇒

equivalent for linear kernel

Kernel PCA

3

Gaussian kernel

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

\( \sigma^2 \) is the "spread"

2

Sigmoid / logistic kernel

\[ k(x, y) = f(x^T y) \]

not really positive semi-definite
\[ f(z) = \frac{2}{c} \frac{e^z}{c^z + 1} \]

Common operations on kernel matrix:
- The corresponding operations in feature space

1. Length of a vector \( \phi(x) \)

\[
\phi(x)^T \phi(x) = k(x,x)
\]

\[
\|\phi(x)\|^2 = k(x,x)
\]

\[
\|\phi(x)\| = \sqrt{k(x,x)}
\]

We do not need \( \phi(x) \) to compute its length!
1. \[ \|\phi(x_i) - \phi(x_j)\|^2 \]

2. \[ = (\phi(x_i) - \phi(x_j))^T (\phi(x_i) - \phi(x_j)) \]

3. \[ = \phi(x_i)^T \phi(x_i) + \phi(x_j)^T \phi(x_j) - 2 \phi(x_i)^T \phi(x_j) \]

4. \[ = k(x_i, x_i) + k(x_j, x_j) - 2k(x_i, x_j) \]

5. **What about the mean in feature space?**

\[ \hat{\mu}_\phi = \frac{1}{n} \sum_{i=1}^{n} \phi(x_i) \]

we do not know \[ \hat{\mu}_\phi \]

\[ \|\phi\|^2 = \hat{\mu}_\phi^T \hat{\mu}_\phi = \left( \frac{1}{n} \sum_{i=1}^{n} \sum_{j=1}^{n} k(x_i, x_j) \right)^n \]
\[ \| \mu \phi \|_2^2 = \mu \phi^T \mu \phi = \left( \frac{1}{n^2} \sum_{i=1}^{n} \sum_{j=1}^{n} k(x_i, x_j) \right) \]

\[ \text{average kernel value} \]

\[ \text{In feature space} \]

\[ = \frac{1}{n} \sum_{i=1}^{n} \frac{1}{\| \phi(x_i) \|^2} \sum_{i=1}^{n} k(x_i, x_i) + \frac{1}{n^2} \sum_{i=1}^{n} \sum_{j=1}^{n} k(x_i, x_j) \]

\[ \frac{1}{n^2} \| \mu \phi \|^2 \]

\[ \sigma_d^2 = \frac{1}{n} \sum_{i=1}^{n} \| \phi(x_i) - \mu \phi \|^2 \]

---

Can we center the data in feature space?

\[ K(x_i, x_j) = \phi(x_i)^T \phi(x_j) \]

before centering

\[ K^C(x_i, x_j) = \]

\[ (\phi(x_i) - \mu \phi)^T (\phi(x_j) - \mu \phi) \]
before centering

\[
\begin{align*}
    k^c(x_i; x_j) &= k(x_i; x_j) - \frac{1}{n} \sum_{a=1}^{n} k(x_i; x_a) - \frac{1}{n} \sum_{a=1}^{n} k(x_j; x_a) + \frac{1}{n^2} \sum_{a=1}^{n} \sum_{b=1}^{n} k(x_a; x_b) \\
    (\phi(x_i) - \mu_\phi) (\phi(x_j) - \mu_\phi) &= \phi(x_i) \phi(x_j) - \phi(x_i) \mu_\phi - \phi(x_j) \mu_\phi + \| \mu_\phi \|^2
\end{align*}
\]

\[
\begin{align*}
    \text{Original} & \quad \Rightarrow \quad \text{centered} \\
    k & \quad \Rightarrow \quad k^c \\
    \text{Axn} & \quad \Rightarrow \quad \text{Axn}
\end{align*}
\]

\[
k^c = 0 \cdot k \cdot 0
\]

\[
0 = I - \frac{1}{n} 1_{n \times n}
\]

(5) Normalization

- project all points \( \phi(x_i) \) onto unit hypersphere
- then compute the kernel
\[ K^n(x_i, x_j) = \left( \frac{\phi(x_i)}{||\phi(x_i)||} \right)^T \left( \frac{\phi(x_j)}{||\phi(x_j)||} \right) \]

\[ \begin{align*}
\text{correlation in feature space} \\
\end{align*} \]

Most operations of interest use only \[ K \]

**Kernel PCA**: non-linear directions that capture the

**Input space**

**Linear direction**

**Feature space**

\[ \phi \]

\[ \phi \text{(quadratic kernel)} \]

\[ \text{line in "quadratic" space} \]
Non-linear dimension

degree 2 curve in input space

\[ \Sigma \tilde{u} = \lambda \tilde{u} \]

usual PCA

\[ \Sigma u = \lambda u \]

usual PCA

\[ \text{Kernel PCA} \]

Input: \( K_{n \times n} \)

1. center \( K \)

\( K^c \)

\[ \Sigma_{\phi} \tilde{u} = \lambda \tilde{u} \]

covariance in feature space

If data has been centered

\[ \Sigma_{\phi} = \frac{1}{n} \sum_{i=1}^{n} \phi(x_i) \phi(x_i)^T \]

\[ \begin{bmatrix} \frac{1}{n} \sum_{i=1}^{n} \phi(x_i) \phi(x_i)^T \end{bmatrix} \tilde{u} = \lambda \tilde{u} \]

\( \tilde{u} = \text{Unknown} \)

\[ \frac{1}{n} \sum_{i=1}^{n} \phi(x_i) a_i = \tilde{u} \]

\[ \sum_{i=1}^{n} \left( \frac{a_i}{\lambda^n} \right) \phi(x_i) = \tilde{u} \]

\[ \sum (c_i \phi(x_i)) = \tilde{u} \]

what \( \tilde{u} \) looks like

Representor theorem

\[ \sum (c_i \phi(x_i)) = \tilde{u} \]
Optimal direction in kernel space is a linear combination of the transformed points.

\[ \sum \phi \hat{u} = \lambda \hat{u} \]

\[
\begin{bmatrix}
\frac{1}{n} \sum_{i=1}^{n} \phi(x_i) \phi(x_i)^T \\
\end{bmatrix} \begin{bmatrix}
\sum_{i=1}^{n} c_i \phi(x_i)
\end{bmatrix} = \lambda \begin{bmatrix}
\sum_{i=1}^{n} c_i \phi(x_i)
\end{bmatrix}
\]

Solve for \(c_i\)'s

\[ \hat{c} = (c_1, c_2, \ldots, c_n)^T \]

\(\hat{c}\) is unknown!

\[ k\hat{c} = \eta_1 \hat{c} \]

\(\hat{c}\) is the dominant eigenvector of \(k\)

\(\eta_1 = n \lambda_1\) is the largest eigenvalue of \(k\)

\(k\): 1. Symmetric \(n \times n\) 2. Positive semi-definite matrix

\(\eta\): sample size
(2) positive semi-definite matrix
\[ \eta_1 \geq \eta_2 \geq \ldots \geq \eta_n \geq 0 \]
\[ \tilde{z} = (c_1, c_2, \ldots, c_n)^T \]
\[ \tilde{u}_i = \sum_{i=1}^{n} c_{i} \phi(x_i) \]
We cannot compute the direction \( \tilde{u}_i \).

Projection of \( \phi(x_0) \) onto \( \tilde{u}_i \)
\[ x_0' = \frac{\phi(x_0)^T \tilde{u}_i}{\tilde{u}_i^T \tilde{u}_i} \]
\[ x_0' = \frac{\phi(x_0)^T \left( \sum_{i=1}^{n} c_{i} \phi(x_i) \right)}{\tilde{u}_i^T \tilde{u}_i} \]

(1) Center \( K \), i.e. \( K_c \)
(2) Compute \( \\begin{vmatrix} K_c & c_i \\ c_i & c_i \end{vmatrix} = 4c_i c_i \) Eigen-decomposition
1. Compute $K \tilde{c}_i = \lambda_i \tilde{c}_i$

2. $\tilde{c}_1, \tilde{c}_2, \ldots, \tilde{c}_n$

3. Scale $\tilde{c}_i$

4. $\tilde{c}_i = \frac{\tilde{c}_i}{\|\tilde{c}_i\|}$

5. Project each pair $x_i$ onto $\tilde{u}_i, \tilde{u}_2, \ldots$

---

**PCA: Unsupervised Learning**

- Directions of new variance

**Linear Discriminant Analysis (LDA)**

- Supervised points are labelled

- Find the direction that maximizes the separation

**Fisher Objective**

- Distance between
Fisher Objective

\[ J(\omega) = \frac{(m_1 - m_2)^2}{s_1^2 + s_2^2} \]

Distance between the projected means

\[ s_i = \text{Scatter of class } i \]
\[ s_i^2 = \frac{1}{n_i} \sum_{i \in C_i} (x_i' - m_i)^2 \]

Scatter total deviation

Variance avg deviation