2 Kernel PCA

2.1 Motivation

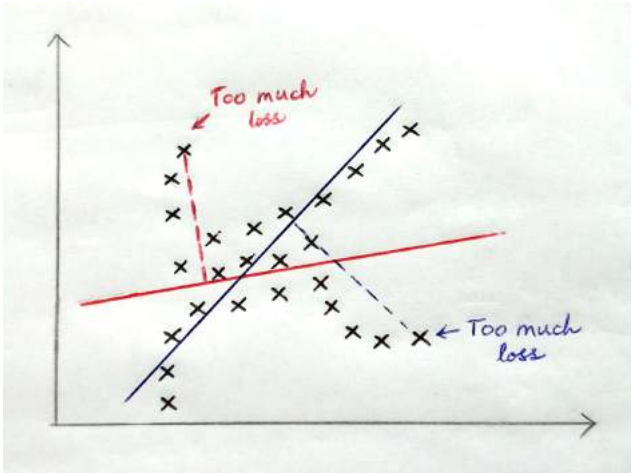

Figure 2: PCA when data is not linearly separable

Dimensionality Reduction, the main objective of Principal Component Analysis, is effective even on highly linearly inseparable data by the use of Kernel PCA.

For example, if 2D data distributed as shown in Figure 2 has to be reduced to 1D, it can be observed that **no single line** exists onto which data can be projected such that the loss of information is minimized (low variance). Here, we need to use **Kernel Trick** that makes the data linearly separable.

The Kernel Trick extends conventional principal component analysis (PCA) to a high dimensional feature space $\phi(x_i)$ in which the data is linearly separable. The PCA determined in this space will then be non-linear in the original data space, which is the desired result.

2.2 Algorithm

KPCA is same as applying PCA on the transformed data $\phi(X)$ instead of X.

1. **Center the data**
$$\hat{x_i} = x_i - \frac{1}{N}\sum_{k=1}^{N} x_k$$

2. **Compute the Sample Covariance Matrix (C)**
$$C = \frac{1}{N}\sum_{i=1}^{N} \phi(\hat{x_i})\phi(\hat{x_i})^T \tag{8}$$

3. **Find the Eigen Decomposition of C**
If C has d eigenvalues and v_i is the eigenvector corresponding to the eigenvalue λ_i,
$$\text{E.D. of } C = \sum_{i=1}^{d} \lambda_i v_i v_i^T$$

4. **Project data onto the first k eigenvectors**

$$\text{Projection of } x_i \text{ on the new system} = \begin{bmatrix} \hat{x_i}^T v_1 \\ \hat{x_i}^T v_2 \\ \vdots \\ \hat{x_i}^T v_k \end{bmatrix}$$

In the above algorithm, steps 2 and 3 (computing C and its eigen decomposition) are never done as we do not know ϕ. Thus, after centering the data, we directly proceed to step 4 to find the projections on the principal components by the method detailed below.

Computing the Projections of Data on the Principal Components:
By the definition of eigenvectors and eigenvalues,

$$Cv_j = \lambda_j v_j \tag{9}$$

$$\therefore \quad v_j = \frac{Cv_j}{\lambda_j}$$

From equation (8),

$$v_j = \sum_{i=1}^{N} \phi(\hat{x_i}) \underbrace{\frac{\phi(\hat{x_i})^T v_j}{\lambda_j N}}$$

Let

$$\frac{\phi(\hat{x_i})^T v_j}{\lambda_j N} = \alpha_{ij} \text{ (a scalar)}$$

Eigenvectors can thus be expressed as a linear combination of features as follows:

$$v_j = \sum_{i=1}^{N} \alpha_{ij} \phi(\hat{x_i}) \tag{10}$$

Although ϕ is unknown, we do know the kernel function K which is a dot product of ϕs (equation (3)). So the goal is to convert our equations to a form involving dot products of ϕs that can be substituted by the kernel function K.

Consider equation (9),

$$Cv_j = \lambda_j v_j$$

From equations (8) and (10),

$$\frac{1}{N} \sum_{k=1}^{N} \phi(\hat{x_k}) \phi(\hat{x_k})^T \sum_{i=1}^{N} \alpha_{ij} \phi(\hat{x_i}) = \lambda_j \sum_{i=1}^{N} \alpha_{ij} \phi(\hat{x_i})$$

$$\frac{1}{N} \sum_{k=1}^{N} \phi(\hat{x_k}) \sum_{i=1}^{N} \alpha_{ij} \underbrace{\phi(\hat{x_k})^T \phi(\hat{x_i})} = \lambda_j \sum_{i=1}^{N} \alpha_{ij} \phi(\hat{x_i})$$

From equation (1),

$$\phi(\hat{x_k})^T \phi(\hat{x_i}) = K(\hat{x_k}, \hat{x_i})$$

$$\Rightarrow \frac{1}{N} \sum_{k=1}^{N} \phi(\hat{x_k}) \sum_{i=1}^{N} \alpha_{ij} K(\hat{x_k}, \hat{x_i}) = \lambda_j \sum_{i=1}^{N} \alpha_{ij} \phi(\hat{x_i})$$

Multiplying with $\phi(\hat{x_l})^T$ on both sides and taking summation over l,

$$\frac{1}{N} \sum_{l=1}^{N} \sum_{k=1}^{N} \underbrace{\phi(\hat{x_l})^T \phi(\hat{x_k})} \sum_{i=1}^{N} \alpha_{ij} K(\hat{x_k}, \hat{x_i}) = \lambda_j \sum_{l=1}^{N} \sum_{i=1}^{N} \alpha_{ij} \underbrace{\phi(\hat{x_l})^T \phi(\hat{x_i})}$$

$$\frac{1}{N}\sum_{l=1}^{N}\sum_{k=1}^{N}K(\hat{x_l},\hat{x_k})\sum_{i=1}^{N}\alpha_{ij}K(\hat{x_k},\hat{x_i}) = \lambda_j\sum_{l=1}^{N}\sum_{i=1}^{N}\alpha_{ij}K(\hat{x_l},\hat{x_i})$$

$$\Rightarrow \quad K^2\alpha_j = N\lambda_j K\alpha_j$$

where

$$\alpha_j = \begin{bmatrix}\alpha_{1j}\\ \alpha_{2j}\\ \vdots\\ \alpha_{Nj}\end{bmatrix}$$

K ($\in \mathbb{R}^{N\text{x}N}$) is called the Kernel Matrix or GRAM MATRIX and is defined as

$$K_{ij} = K(x_i, x_j) \tag{11}$$

K has the property of always being invertible. Multiplying both sides with K^{-1} will only affect the eigenvectors with zero eigenvalue, which will not be a principle component anyway.

$$\Rightarrow \quad K\alpha_j = (N\lambda_j)\alpha_j \tag{12}$$

The above equation is equivalent to an eigen decomposition of the matrix K with j^{th} eigenvalue = $N\lambda_j$ and corresponding eigenvector = α_j.

In equation (8), as $\phi(\hat{x_i})$ is not known, the exact value of v_j cannot be determined. But what we need is the projection of a (centered and transformed) data point on the k principal components v_j, which can be calculated as follows:

$$\phi(\hat{x_k})^T v_j = v_j^T\phi(\hat{x_k}) = \sum_{i=1}^{N}\alpha_{ij}\underbrace{\phi(\hat{x_i})^T\phi(\hat{x_k})}$$

So given any new data point x_k, find $\hat{x_k}$. Then determine its projection onto the k principal components as,

$$\Rightarrow \quad \text{Projection of } \hat{x_k} = \sum_{i=1}^{N}\alpha_{ij}K(\hat{x_i},\hat{x_k}) \tag{13}$$