Lecture 5 Sept. 22-2006

Data Visualization

STAT 442 / 890, CM 462

Lecture: Ali Ghodsi

1 Alternative Derivation

Another nice property of PCA, closely related to the original discussion by Pearson [1], is that the projection onto the principal subspace minimizes the squared reconstruction error, $\sum_{i=1}^{t} ||x_i - \hat{x}_i||^2$. In other words, the principal components of a set of data in $\Re^n$ provide a sequence of best linear approximations to that data, for all ranks $d \leq n$.

Consider the rank-d linear approximation model as :

$$f(y) = \bar{x} + U_d y$$

This is the parametric representation of a hyperplane of rank d.

For convenience, suppose $\bar{x} = 0$ (otherwise the observations can be simply replaced by their centered versions $\tilde{x} = x_i - \bar{x}$). Under this assumption the rank d linear model would be $f(y) = U_d y$, where U_d is a $n \times d$ matrix with d orthogonal unit vectors as columns and y is a vector of parameters. Fitting this model to the data by least squares leaves us to minimize the reconstruction error:

$$\min_{U_d, y_i} \sum_{i}^{t} ||x_i - U_d y_i||^2$$

By partial optimization for y_i we obtain:

$$\frac{d}{dy_i} = 0 \Rightarrow y_i = U_d^T x_i$$

Now we need to find the orthogonal matrix U_d:

$$\min_{U_d} \sum_i^t ||x_i - U_d U_d^T x_i||^2$$

Define $H_d = U_d U_d^T$. H_d is a $n \times n$ matrix which acts as a projection matrix and projects each data point x_i onto its rank d reconstruction. In other words, $H_d x_i$ is the orthogonal projection of x_i onto the subspace spanned by the columns of U_d. A unique solution U can be obtained by finding the singular value decomposition of X [2]. For each rank d, U_d consists of the first d columns of U.

2 Toward Kernel PCA

PCA is designed to model linear variabilities in high-dimensional data. However, many high dimensional data sets have a nonlinear nature. In these cases the high-dimensional data lie on or near a nonlinear manifold (not a linear subspace) and therefore PCA can not model the variability of the data correctly. One of the algorithms designed to address the problem of nonlinear dimensionality reduction is Kernel PCA (See Figure 1 for an example). In Kernel PCA, through the use of kernels, principle components can be computed efficiently in high-dimensional feature spaces that are related to the input space by some nonlinear mapping.

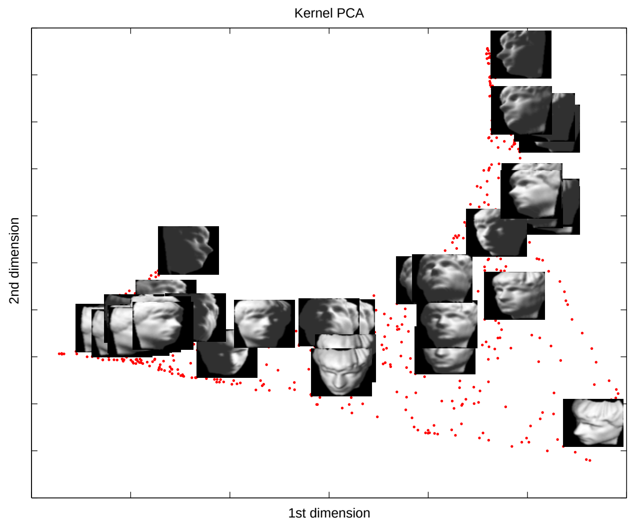

Figure 1: *Kernel PCA with Gaussian kernel applied to the same data set. A two-dimensional projection is shown, with a sample of the original input images.*

Kernel PCA finds principal components which are nonlinearly related to the input space by performing PCA in the space produced by the nonlinear mapping, where the low-dimensional latent structure is, hopefully, easier to discover.

Example:

The usage of kernel can be best seen through a classification example. Consider the diagram (Fig. 2). In the two dimensional space the separation boundary is not linear. In other words, it is impossible to separate dots from crosses by a liner classifier (a line in two-dimensional space). Now suppose an auxiliary dimension is added to this 2-dimensional space. More precisely suppose any 2-dimensional point $\vec{x}$ is mapped to a 3-dimensional point $\vec{\Phi x}$ (we call this new space, feature space) as follows:

$$\begin{pmatrix} x_1 \\ x_2 \end{pmatrix} \vdash \begin{pmatrix} x_1^2 \\ x_2^2 \\ \sqrt{2}x_1x_2 \end{pmatrix}$$

It is clear that in the 3-dimensional space the points become linearly separable. So that they can indeed be separted by a linear hyperplane. We can take advantage of the fact that in higher dimensions the separation may become linear.

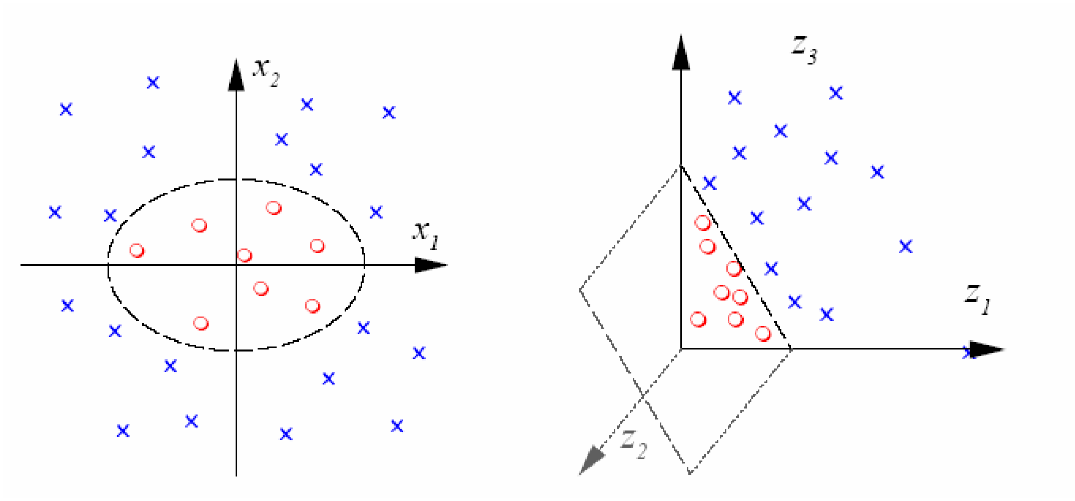

Figure 2: *Two dimensional classification example. A separation in feature space can be found using a linear hyperplane (right). In input space this construction corresponds to a non-linear ellipsoidal decision boundary (left). (figure from Scholkopf and Smola 2002)*

Working with points in feature space (a space with higher dimension) will increase the computational complexity and therefore will not be desirable. However, there is a computational shortcut which makes it possible to represent linear structure efficiently in high-dimensional spaces. The shortcut is what we call a kernel function.

Definition 1: [Kernel function] A kernel is a function k that for all $x, z \in X$ satisfies

$$K(x,z) \quad = \quad <\Phi(x), \Phi(z)>$$

where $< .,. >$ is inner product and Φ is a mapping from X to an feature space.

Example 1: Consider a two-dimensional input space together with the feature map

$\Phi : \mathbf{x} = (x_1, x_2) \vdash \Phi(\mathbf{x}) = (x_1^2, x_2^2, \sqrt{2}x_1x_2) \in \mathcal{H}$.

The inner product in the feature space can be evaluated as follows:

$$\begin{aligned} <\Phi(\mathbf{x}), \Phi(\mathbf{z})> &= \Big\langle x_1^2, x_2^2, \sqrt{2}x_1x_2), z_1^2, z_2^2, \sqrt{2}z_1z_2) \Big\rangle \\ &= x_1^2z_1^2 + x_2^2z_2^2 + 2x_1x_2z_1z_2 \\ &= (x_1z_1 + x_2z_2)^2 = <\mathbf{x}, \mathbf{z}>^2 \end{aligned}$$

Hence, the function

$$k(\mathbf{x}, \mathbf{z}) \quad = \quad <\mathbf{x}, \mathbf{z}>^2$$

is a kernel function with $\mathcal{H}$ its corresponding feature space.

The following is a short list of common kernels.

Linear Kernel:

$$k_{ij} = <x_i, x_j>$$

Gaussian Kernel:

$$k_{ij} = e^{\frac{-||x_i - x_j||^2}{2\sigma^2}}$$

Polynomial Kernel:

$$k_{ij} = (1 + <x_i, x_j>)^p$$

References

[1] K. Pearson. On lines and planes of closest fit to systems of points in space. *The London, Edinburgh and Dublin Philosophical Magazine and Journal of Science*, Sixth Series 2:559–572, 1901.

[2] J. Friedman T. Hastie, R. Tibshirani. *The elements of statistical learning*. Springer, New York, 2002.