Exercises
Machine Learning
AS 2012

Machine Learning Laboratory
Dept. of Computer Science, ETH Zürich
Prof. Dr. Joachim M. Buhmann
Web http://ml2.inf.ethz.ch/courses/ml/

Email questions to: Sharon Wulff
sharon.wulff@inf.ethz.ch

Series 7, Dec 18th, 2012 (Kernel PCA, K-means)

Problem 1 (Kernel PCA):

Two of the methods, which we have seen during the course, Principal Component Analysis (PCA) and the kernel method, can be combined into a machine learning algorithm known as *kernel PCA*. The lesson learned from kernelized support vector machines seems to be that many complex structures in data can be linearized by mapping into high-dimensional space (by means of the kernel trick). PCA is a very useful and widely applied linear method, so can we apply the kernel trick to PCA?

Given is a set of data in low-dimensional space, $\mathbf{x}_1, ..., \mathbf{x}_n \in \mathbb{R}^L$. In addition, we have a kernel k, which implicitly defines a mapping ϕ into a high-dimensional space $\mathbb{R}^H$:

$$\begin{aligned} k(\mathbf{x}, \mathbf{y}) &= \langle \phi(\mathbf{x}) | \phi(\mathbf{y}) \rangle_{\mathbb{R}^H} \\ \phi : \mathbb{R}^L &\rightarrow \mathbb{R}^H \end{aligned} \tag{1}$$

$\langle \, . \, | \, . \, \rangle_{\mathbb{R}^H}$ is the standard scalar product on $\mathbb{R}^H$. Once again, recall the kernel assumption that we can compute $k(\mathbf{x}, \mathbf{y})$, but *not* $\phi(\mathbf{x})$. What we would like to do (if we could handle objects in $\mathbb{R}^H$) is:

1. Map the data into the high-dimensional space $\mathbb{R}^H$.
2. Apply standard PCA in $\mathbb{R}^H$. First step: Compute the empirical covariance matrix of the (mapped) data $\phi(\mathbf{x}_i)$ into $\mathbb{R}^H$:
$$\Sigma^H := \hat{\mathrm{E}}[\phi(\mathbf{x})\phi(\mathbf{x})^t] \in \mathbb{R}^{H \times H} \tag{2}$$
3. Second step: To obtain an N-dimensional PCA projection, compute the first N eigenvectors of Σ:
$$\mathbf{v}_1, ..., \mathbf{v}_N \tag{3}$$
4. Finally: Compute the projection $P(\phi(\mathbf{x}_i)) \in \mathbb{R}^N$ for each data point. The j-th component of the projection is given by
$$P_j(\phi(\mathbf{x}_i)) = \langle \phi(\mathbf{x}_i) | \mathbf{v}_j \rangle_{\mathbb{R}^H} \ . \tag{4}$$

Note that we can choose the number of embedding dimensions N (which will usually be small, like $N = 2$ or $N = 3$); but the vectors $\phi(\mathbf{x}_i)$ and the eigenvectors $\mathbf{v}_i$ are of length H, and the size of the covariance matrix is $H \times H$. Even if H is finite, computing eigenvectors will be an $O(H^3)$ operation. To perform the above algorithm, we will have to figure out a way to perform each of the steps by means of the kernel trick, including the eigen-decomposition of Σ^H. This exercise problem will take you through the derivation of the algorithm in a step-by-step manner.

Reference formulæ

Recall that an empirical covariance matrix is a superposition of outer products of the form $\mathbf{x}\mathbf{x}^t$:

$$\Sigma = \hat{\mathrm{E}}[\mathbf{x}\mathbf{x}^t] = \frac{1}{n} \sum_{k=l}^{n} \mathbf{x}_l \mathbf{x}_l^t \ . \tag{5}$$

(To keep things simple, we have assumed that the data is centered, so we do not have to subtract the mean when computing the variance.) The outer product $\mathbf{x}\mathbf{x}^t$ of a vector with itself is a matrix with elements $(\mathbf{x}\mathbf{x}^t)_{ij} = x_i x_j$. As you will recall from the lectures on SVMs, the kernel matrix K is defined by

$$K_{ij} := k(\mathbf{x}_i, \mathbf{x}_j) = \langle \phi(\mathbf{x}_i) | \phi(\mathbf{x}_j) \rangle_{\mathbb{R}^H} \ . \tag{6}$$

We denote the eigenpairs (eigenvalues and eigenvectors) of Σ^H by $(\lambda_l, \mathbf{v}_l)$, and those of K by $(\rho_l, \mathbf{w}_l)$. Thus:

$$\begin{aligned} \Sigma^H \mathbf{v}_i &= \lambda_i \mathbf{v}_i \qquad \text{for } i = 1, ..., H \\ K \mathbf{w}_j &= \rho_j \mathbf{w}_j \qquad \text{for } j = 1, ..., n. \end{aligned} \tag{7}$$

The *data matrix* of the sample is the matrix containing the sample points as its rows:

$$\mathbf{X} := \begin{pmatrix} \mathbf{x}_1^t \\ ... \\ \mathbf{x}_n^t \end{pmatrix} \in \mathbb{R}^{L \times n} \tag{8}$$

The corresponding data matrix of the mapped data in $\mathbb{R}^H$ will be denoted Φ:

$$\Phi := \begin{pmatrix} \phi(\mathbf{x}_1)^t \\ ... \\ \phi(\mathbf{x}_n)^t \end{pmatrix} \in \mathbb{R}^{H \times n} . \tag{9}$$

Problems

Please show the following:

1. Show that an empirical covariance matrix Σ can be rewritten as

$$\Sigma = \frac{1}{n} \mathbf{X}^t \mathbf{X} . \tag{10}$$

   Obviously, the same will then hold on $\mathbb{R}^H$:

$$\Sigma^H = \frac{1}{n} \Phi^t \Phi . \tag{11}$$

2. Show that the kernel matrix K can be written as

$$K = \Phi \Phi^t . \tag{12}$$

   (We say that Σ^H and K are *dual* up to the coefficient $\frac{1}{n}$.) What are the sizes of Σ^H and K?

3. Assume that $(\rho_j, \mathbf{w}_j)$ is an eigenpair of K. Use (11), (12) and the eigenvalue equations (7) to show that $\mathbf{v}_j := \Phi^t \mathbf{w}_j$ is an eigenvector of Σ^H and find corresponding eigenvalue λ_j.

4. With the above result, we can compute eigenvectors of Σ^H from eigenvectors of K, but the $\mathbf{v}_j$ are not normalized (no unit vectors). Show that the unit vector $\tilde{\mathbf{v}}_j$ (with respect to the norm defined by $\langle . | . \rangle_{\mathbb{R}^H}$) in the same direction is given by $\tilde{\mathbf{v}}_j := \frac{1}{\sqrt{\rho_j}} \mathbf{v}_j$.

5. Assume that we know the eigenvalues of K, have selected the N largest ones and computed the corresponding eigenvectors $\mathbf{w}_1, ..., \mathbf{w}_N$. We are given a point $\mathbf{x} \in \mathbb{R}^L$. Derive a formula which computes the projection $P(\phi(\mathbf{x}))$ in (4) onto $\tilde{\mathbf{v}}_1, ..., \tilde{\mathbf{v}}_N$ using *only* the kernel and objects defined in $\mathbb{R}^L$. Please ensure that the final projection formula does not explicitly contain ϕ or any vector in or matrix on $\mathbb{R}^H$.

6. Finally, summarize: Rewrite the (infeasible) algorithm on the previous page in terms of the derived results, as a method that could actually be implemented. The input are the samples $\mathbf{x}_1, ..., \mathbf{x}_n$, the kernel k and the number N of projection dimensions. The output are the projected data points $(P_1(\phi(\mathbf{x}_i)), ..., P_N(\phi(x_i)))^t$ for $i = 1, ..., n$.

Problem 2 (K-means):

Show that the k-means algorithm (for Euclidean distance) will always converge. In the lecture the algorithm was directly presented without discussing the explicit cost function that k-means is minimizing. We will show that the algorithm optimizes the following cost function

$$J = \sum_{n=1}^{N} \sum_{k=1}^{K} r_{nk} \|\mathbf{x}_n - \mu_k\|^2 .$$

Where $r_{nk} \in \{0, 1\}$ with $\sum_{k=1}^{K} r_{nk} = 1$ and $\mu_k = \frac{\sum_{n=1}^{N} r_{nk} \mathbf{x}_n}{\sum_{n=1}^{N} r_{nk}}$.

1. How would you choose r_{nk} to minimize J for given μ_k? Note that this corresponds to the Assignment step of the k-means algorithm.
2. Compute $\frac{\delta J}{\delta \mu_k}$ and set it to zero. You should identify the Update step of k-means.

As both steps of the algorithm decrease the value of J, you've just proven, that the algorithm always converges.