Machine Learning
Exercise 4

Marc Toussaint
Machine Learning & Robotics lab, U Stuttgart
Universitsstrae 38, 70569 Stuttgart, Germany

May 24, 2013

1 PCA and reconstruction on the Yale face database

On the webpage find and download the Yale face database http://ipvs.informatik.uni-stuttgart.de/mlr/marc/teaching/data/yalefaces.tgz. (Optinally use `yalefaces_cropBackground.tgz`.) The file contains gif images of 166 faces.

a) Write a routine to load all images into a big data matrix $X \in \mathbb{R}^{165\times 77760}$, where each row contains a gray image.

In Octave, images can easily be read using `I=imread("subject01.gif");` and `imagesc(I);`. You can loop over files using `files=dir(".");` and `files(:).name`.

b) Compute the mean face $\mu = 1/n \sum_i x_i$ and center the whole data, $X \leftarrow X - \mathbf{1}_n \mu^\top$.

c) Compute the singular value decomposition $X = UDV^\top$ for the data matrix.[1] In Octave/Matlab, use the command `[U, S, V] = svd(X, "econ")`, where the `econ` ensures we don't run out of memory.

d) Map the whole data set to $\boldsymbol{Z} = \boldsymbol{X} V_p$, where $V_p \in \mathbb{R}^{77760\times p}$ contains only the first p columns of V. Assume $p = 60$. The Z represents each face as a p-dimensional vector, instead of a 77760-dimensional image.

e) Reconstruct all images by computing $\tilde{\boldsymbol{X}} = \mathbf{1}_n \mu^\top + \boldsymbol{Z} V_p^\top$. Display the reconstructed images (by reshaping each row of $\tilde{\boldsymbol{X}}$ to a 320×243-image) – do they look ok? Repeat for various PCA-dimensions $p = 1, 2, ...$

2 PLS for classification?

In the course we discussed Partial Least Squares as a method that, instead of just picking the p largest PCA components, it incrementally picks those components that are most correlated *with the output*. Meassuring correlation with the output is obvious in the regression case ($\widehat{\boldsymbol{X}}^\top \boldsymbol{y}$ in the algorithm).

Do research to generalize PLS to the classification case. Report on your ideas, what you find in the web, and a pseudocode describing such an PLS-for-classification method.

[1]This is alternative to what was discussed in the lecture: In the lecture we computed the SVD of $X^\top X = (UDV^\top)^\top (UDV^\top) = VD^2V^\top$, as U is orthonormal and $U^\top U = \mathbf{I}$. Decomposing the covariance matric $X^\top X$ is a bit more intuitive, decomposing X directly is more efficient and amounts to the same V.