TECHNISCHE UNIVERSITÄT MÜNCHEN
Fakultät für Elektrotechnik und Informationstechnik
Lehrstuhl für Datenverarbeitung
Prof. Dr. Martin Kleinsteuber

`Information Retrieval in High Dimensional Data`

`Lab #7`, 7.12.2017

Linear Discriminant Analysis

Task 1. In this task, we will once again work with the MNIST training set as provided on Moodle. Choose three digit classes, e.g. 1, 2 and 3 and load `N=1000` images from each of the classes to the workspace. Store the data in a normalized matrix `X` of type size `(784,3*N)`. Furthermore, generate a color label matrix `C` of dimensions `(3,3*N)`. Each row of `C` assigns an RGB color vector to the respective column of X as an indicator of the digit class. Choose `[0, 0, 1]`, `[0, 1, 0]` and `[1, 0, 0]` for the three digit classes.

a) Compute the principal subspace `U` of dimension `2` of `X`. Create a `C`-colored scatter plot of the scores of `X` with respect to this subspace.

b) Generate a matrix `X_sums` of size `(784,3)` which consists of the row-wise sums of the centered data samples belonging to each of the three classes, normalized by the square roots of the class sizes. From `X_sums`, compute the principal subspace `U_b` of dimension `2` and generate a `C`-colored `scatter` plot of the scores of `X` with respect to `U_b`. Compare the plot with the one from a). Which representation would you choose for a k-Nearest-Neighbors classification?

c) Write a PYTHON function `sqrtminv` which expects a symmetric positive definite matrix `A` as its input and returns the inverse of its square root as its output, without using `scipy.linalg.sqrtm`.

d) Divide `X` into three matrices of sizes `(784,N)`, each containing the samples belonging to one of each of the classes. Center the three matrices to create `X1c`, `X2c` and `X3c` and compute the within-class-center matrix `S_w=(np.dot(X1c,X1c.T)+np.dot(X2c,X2c.T)+np.dot(X3c,X3c.T))`.

e) Calculate the "normalized class sum matrix" `S_bw=np.dot(sqrtminv(S_w),X_sums)` and the left singlar vectors `U_bw` corresponding to its `2` largest singular values. Create a `C`-colored `scatter` plot of the scores of `X` with respect to the subspace described by `sqrtminv(S_w)*U_bw` (Hint: orthogonolize the basis vectors). Compare the plot with the ones from a) and b). Which representation would you choose for a k-Nearest-Neighbor classification? What weaknesses does this implementation of LDA have?

Helpful Python/Numpy functions

`np.linalg.qr(A)`	Returns the QR orthogonalization of `A`
`np.sqrt(X)`	Square root