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

Information Retrieval in High Dimensional Data

Lab #9, 18.01.2017

Kernel PCA

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=500 images from each of the classes to the workspace. Store the data in a normalized matrix `X_mnist` of type double and size `([784,3*N)`. Furthermore, generate a color label matrix `C` of dimensions `(3,3*N)`. Each column of `C` assigns an RGB color vector to the respective column of `X_mnist` 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_mnist`. Create a `C`-colored scatter plot of the scores of `X_mnist` with respect to this subspace.

b) Write a Python function `kgram` which expects a data matrix `X` of size `(p,N_X)` and a kernel function handle `kappa` as its input. It returns the Gram matrix `K` of `X` with respect to `kappa`. In order to reduce the number of for loops, assume that `kappa` accepts matrices as its input and calculates the kernel column-wise, returning a row vector as its output.

c) Write a Python function `kpca` which expects a data matrix `X`, a kernel function handle `kappa` and the dimension of the intrinsic subspace `k` as its input, and returns the Kernel PCA scores `S` of `X`. The representation is to be computed according to the equation (7.11) in the lecture notes.

d) Generate scatter plots of the scores produced by `kpca(X_mnist,kappa,k)`. Choose `k=3` and[1]

- `def kappa(X,Y): return np.sum(X*Y, axis=0)`
- `def kappa(X,Y): return (np.sum(X*Y, axis=0)+c)**d`
- `def kappa(X,Y):`
  `return np.exp(-np.sum((X-Y)**2, axis=0)/(2*sigma**2))`

as kernel functions. What are the names of the kernels? Try out different values for `c`, `d` and `sigma`. Refer to the lecture notes for the value ranges.

[1] `X` and `Y` are appropriate matrices

Helpful Python/Numpy functions

For 3D scatter plots use the following commands

```
from mpl_toolkits.mplot3d import Axes3D
import matplotlib.pyplot as plt
fig = plt.figure()
ax = fig.add_subplot(111, projection='3d')
ax.scatter(x, y, z, c='b', marker='o')
```