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

Information Retrieval in High Dimensional Data

Assignment #3, 19.06.2018

Due date: 06.07.2018, 6 P.M.

Please hand in your solutions via Moodle. Use the attached Jupyter notebook.

Solutions must be handed in by groups. Please state the names of your group members at a prominent place in your submission. (For example, at the beginning of your provided notebook or in a separate text file.)

The Kernel Trick

Task 1: [25 points] On Moodle you will find a Jupyter-Notebook that contains a function for dimensionality reduction via PCA. The function `linear_pca` expects a data matrix $\mathbf{X} \in \mathbb{R}^{p \times N}$ and a number of PCs k and returns the first k PCA scores for the matrix $\mathbf{X}$.

- Provide code that tests the function with selected images from the provided MNIST training dataset by visualizing the first 2 scores in a scatter plot.
- Complete the function `gram_pca` such that it has the same functionality as `linear_pca` but expects a gram matrix $\mathbf{K} = \mathbf{X}^\top \mathbf{X}$ instead of the data matrix $\mathbf{X}$ as its input. Do not assume that $\mathbf{K}$ was produced from centered data. Note: It is important to be consistent in notation here. E.g., for a data matrix of 1000 MNIST images, we have $\mathbf{X} \in \mathbb{R}^{784 \times 1000}$ and $\mathbf{K} \in \mathbb{R}^{1000 \times 1000}$.
- Test your implementation and show that `gram_pca(dot(X.T,X), k)` yields results equivalent to those of `linear_pca(X, k)`.
- There is as an unknown vector space $\mathbb{H}$, equipped with an inner product $\langle \cdot, \cdot \rangle_{\mathbb{H}}$ and a function
$$\varphi : \mathbb{R}^p \to \mathbb{H},$$
such that
$$\langle \varphi(\mathbf{x}), \varphi(\mathbf{y}) \rangle_{\mathbb{H}} = \exp\left(-\frac{\|\mathbf{x} - \mathbf{y}\|^2}{2\sigma^2} \right)$$
holds for every $\mathbf{x}, \mathbf{y} \in \mathbb{R}^p$. The expression on the right-hand side of the equation is called the *Gaussian kernel* and σ is a parameter to choose by hand.

The function `gaussian_kernel_pca` expects a data matrix $\mathbf{X}$, a reduced dimension number k and a parameter σ. It returns the first k *Kernel PCA* scores of the data. In other words, the function returns the first k PCA scores of

$$\varphi(\mathbf{x}_1), \varphi(\mathbf{x}_2), \ldots, \varphi(\mathbf{x}_N),$$

where $\mathbf{x}_i$ denotes the i-th data sample/i-th column of the data matrix. The function `gaussian_kernel_pca` is already written, but for it to work, the function `compute_gaussian_gram_matrix` must return correct results. Complete `compute_gaussian_gram_matrix` accordingly.

- Test `gaussian_kernel_pca` with some MNIST train images and $\sigma = 1000$.