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

`Information Retrieval in High Dimensional Data`

`Lab #10`, 25.01.2018

CVXOPT

Task 1. Machine Learning tasks are typically thought of optimization problems, e.g. minimizing an error function or maximizing a probability. Ideally, the optimization problem turns out to be convex, which implies that any local minimum is the global minimum of the formulation. In the following, it will be assumed that you have some basic knowledge about convex optimization. The intention of this task is to familiarize ourselves with CVXOPT, one of the most-widely used convex optimization toolboxes. Note: If CVXOPT does not accept your NumPy arrays, try casting them to `double`.

a) Go to `cvxopt.org` and follow the installation instructions for your distribution. For conda, you need to run

```
conda install -c conda-forge cvxopt
```

b) Skim through the **Examples** section on `cvxopt.org` to get an overview of the functionality of the different solvers of CVXOPT.

c) Implement a function `minsq` which expects a NumPy array `A` of shape `(m,n)` and a NumPy array `y` of shape `(m,)` as its arguments and returns a NumPy array `x` of shape `(n,)` that solves the following problem.

$$\min_{\mathbf{x}} \|\mathbf{A}\mathbf{x} - \mathbf{y}\|.$$

Test your function by feeding it with appropriate inputs and comparing the results with the ones you get by using `np.linalg.pinv`. Experiment by adding white Gaussian noise to `y`.

d) Consider the equation (8.30) in the lecture notes. Implement a function `solvedualsvm(H,y)` that returns the solution `lambda_star` of the dual SVM problem by means of CVXOPT. Test your function with the training data

$$\mathbf{x}_1 = \begin{bmatrix} -1 \\ -1 \end{bmatrix}, y_1 = -1,\ \mathbf{x}_2 = \begin{bmatrix} -2 \\ -2 \end{bmatrix}, y_2 = -1,$$
$$\mathbf{x}_3 = \begin{bmatrix} 1 \\ 1 \end{bmatrix}, y_3 = 1,\ \mathbf{x}_4 = \begin{bmatrix} 2 \\ 2 \end{bmatrix}, y_4 = 1,$$

Verify that the KKT conditions with respect to the support vectors are in line with what you expect. In the next lab course, we will use this function to implement linear and kernel SVM.

Helpful Python/Numpy functions

`from cvxopt import matrix, solvers`	Basic CVXOPT functionality
`solvers.qp`	Quadratic Programming
`numpy.ndarray.astype`	Array casting (use `'double'`)