Lab Course Machine Learning and Data Analysis (Praktikum Maschinelles Lernen und Datenanalyse)

Fachgebiet Maschinelles Lernen
Fakultt IV, Technische Universitt Berlin
Prof. Dr. Klaus-Robert Mller
Email: klaus-robert.mueller@tu-berlin.de

Summer term 2016

Problem set 4: Support Vector Machines

Part 1: Implementation

Assignment 1 (15 points)

Implement the SMO algorithm for SVMs. Refer to the handbook for pseudo code. The signature follows that of kernel ridge regression. Implement a class

```
svm_smo(kernel, kernelparameter, regularization)
```

with functions `fit(X, y)` and `predict(X)`. The labels in y will be +1/-1. Your `fit` method should only save the support vectors, not all data points. The class has three subfunctions for the SMO algorithm:

- `L, H = _compute_box_constraints(self, i, j, Y, alpha, C)`
- `new_alpha, new_b, updated = _update_parameters(self, E_i, E_j, i, j, K, Y, alpha, b, C)` where updated is `True` or `False`.
- `new_b = _compute_updated_b(self, E_i, E_j, i, j, K, Y, alpha_old, alpha_new, b_old, C)`.

The test script provided will test these functions to make potential debugging easier.
Hint: If you do predictions, you might want to use the `sign` function to get +1/-1 labels.

Assignment 2 (5 points)

Implement a plotting method

```
plot_svm_2d(X, y, model)
```

that takes as input a $(n \times 2)$ array `X`, +1/-1 labels `y` and a fitted svm model. It should plot the points in `X` as circles in different colors, mark support vectors with a cross *and plot the separating hyperplane.*
Hint: To plot the separating hyperplane evaluate your classifier on each point of a grid that stretches all data points and then plot the contour line.

Assignment 3 (10 points)

Write the SVM dual optimization problem as a quadratic programming (QP) problem in the form of

$$\begin{aligned} \min_x \quad & \frac{1}{2} x^T P x + q^T x \\ \text{s.t.} \quad & Gx \preceq h \\ & Ax = b \end{aligned}$$

Describe the variables in your report. Implement this in the class `svm_qp` with the help of `cvxopt.solvers.qp` (see stubs). The class should have the same signature and functions as `svm_smo`. You might have to install the `cvxopt` package via `pip install cvxopt`.

Part 2: Application

Assignment 4 (5+10+10=25 points)

Use your SVM SMO implementation to train classifiers on the `easy_2d` dataset from the ISIS site.

1. Find the optimal parameters C and σ for a Gaussian kernel and plot the results. Use your cross validation method. If you do not have a running cross validation method, contact us.
2. Train one model for a σ and C that obviously overfit and for a σ and C that obviously underfit the data. Plot the results.
3. For optimal C and σ, plot a receiver operator characteristics (ROC) curve by varying the bias parameter b of your SVM model.

Assignment 5 (10 points)

Besides from an implementation in a low level language like C, how would you optimize your SMO implementation? I.e. where in the algorithm described in the handbook/guide do you see room for improvement?

Assignment 6 (15 points)

In this assignment you will work on the UCI Iris dataset[1]. This is a 4-dimensional dataset with 150 instances in 3 classes. Use the `.npy` file from the ISIS site. Which classes are linearly separable from the two other classes and which classes are not? Are they separable with a non-linear classifier? Describe what you tested. Provide the found hyperparameters and classification accuracies.

Assignment 7 (10+5+5=20 points)

We would now like to use the SVM on a 'real' dataset, the USPS dataset, which we saw on earlier sheets. In this case, one trains to classify one digit against the rest of the digits (one-vs-rest classification).

1. Use 5-fold Cross Validation (see exercise sheet 4), to find the best kernel and kernel parameters for each digit. Report the best kernel and kernel parameter as well as the estimated test error for each digit.
2. Plot, for each kernel, 5 randomly chosen Support Vectors from the two classes.
3. What problem do we have when using one-vs-rest classification with the SVM implemented via the guide book pseudo-code (that would not exist with one-vs-one classification)? How could we change the SVM method to fix this? (You only have to describe it, you do not have to implement it.)

[1] http://archive.ics.uci.edu/ml/datasets/Iris