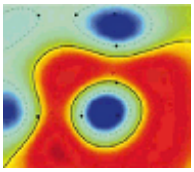
Technische Universität Berlin



Problem Set 4: Support Vector Machines

Report Machine Learning Lab Course
Fachgebiet Maschinelles Lernen
Prof. Dr. Klaus-Robert Müller
Fakultät IV Elektrotechnik und Informatik
Technische Universität Berlin

submitted by

Budi Yanto

Instructor: Daniel Bartz
Felix Brockherde

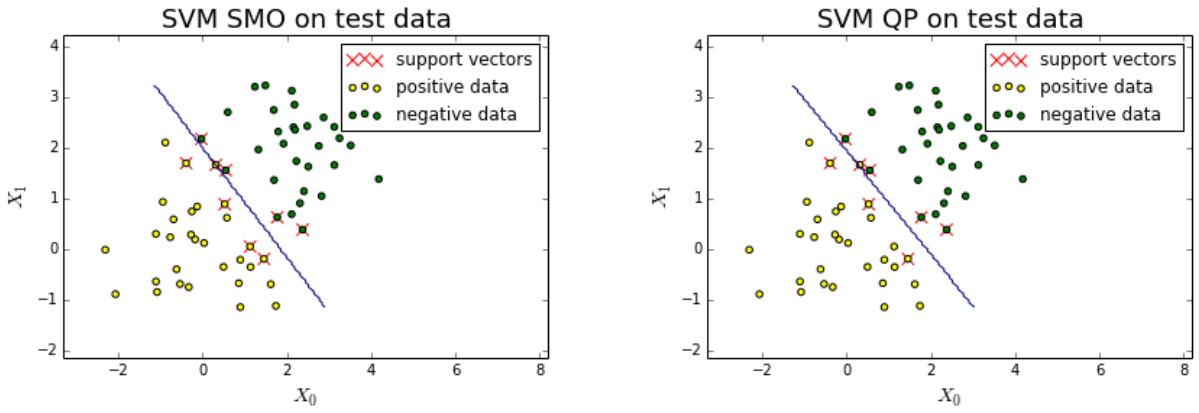
Matrikelnummer: 308819
Email: budiyanto@mailbox.tu-berlin.de

Part 1: Implementation

In this part, the pseudocode of the SMO from the handbook and the class *svm_qp* that solves the SVM dual optimization problem as a quadratic programming problem were implemented. Furthermore, a function to plot the SVM 2D was also implemented.

Assignment 1: SMO

The SVM SMO should be implemented in this assignment. The implementation was pretty straight forward and the pseudocode in the handbook was really helpful to implement the SMO. All of the tests provided for this assignment were passed. Figure 1a shows the 2D plot of the SVM SMO applied to the test data.



(a) SVM SMO applied to the test data

(b) SVM QP applied to the test data

Figure 1: SVM SMO and SVM SMO applied to the test data

Assignment 2: Plot SVM 2D

This function was implemented to plot the 2D data points and its corresponding support vectors. In addition, the separating hyperplane is also plotted. The positive data points are drawn as yellow circles whereas the negative data points are drawn as green circles. The support vectors on the other hand are drawn as red crosses.

Assignment 3: SVM QP

In this assignment, the SVM dual optimization problem should be solved as a quadratic programming problem. The method *cvxopt.solvers.qp* from the package *cvxopt* was used to solve the QP problem. It was pretty straight forward to implement this method. We have to figure out how to reformulate the SVM dual optimization problem to the form of QP problem. The most difficult part in this assignment was actually trying to figure out how to calculate the bias b. Figure 1b shows the 2D plot of the SVM QP applied to the test data.

Part 2: Application

In this part, the SVM SMO implementation should be applied to the *Easy_2D* and *UCI Iris* datasets. In addition to that, the running time of the *svm_qp* implementation should be compared to the properly implemented SMO routine *svm_sklearn*.

Assignment 4: Easy_2D Dataset

In this assignment, the Cross-Validation should be used to find the optimal parameters C and σ for a Gaussian kernel. Table 1 shows the parameters used to run the Cross-Validation whereas Table 2 shows the optimal parameters and properties obtained by the Cross-Validation. The result of the SVM that run using the optimal parameters is depicted in Figure 2. The SVM SMO obtained 55 support vectors from total 100 data points. The SVM SMO with the optimal parameters was also run on the test data from the easy_2D dataset, and got 0.01 loss which is actually quite low. It got 0.07 loss when it was run to predict the true data.

Kernel	Kernel Parameters	Regularization	nRepetitions	nFolds
Gaussian	np.logspace(-2,2,10)	np.logspace(-2,2,10)	2	10

Table 1: Parameters used to run the Cross-Validation

Kernel	Kernel Parameters	Regularization	Cvloss	Loss on easy_Xte
Gaussian	0.21544	1.6681	0.1	0.01

Table 2: Optimal parameters and properties obtained by the Cross-Validation

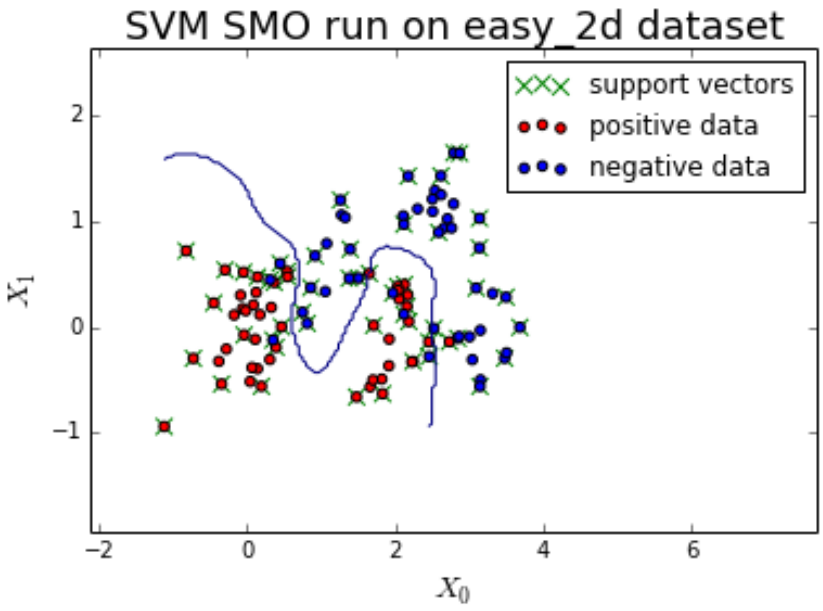
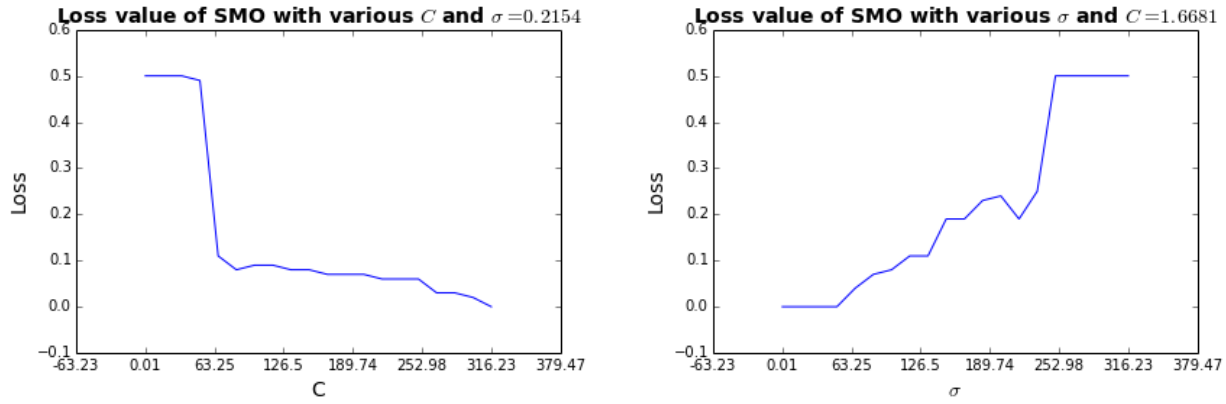


Figure 2: SVM SMO applied to the *easy_2D* dataset

Now the SVM should be trained with C and σ that obviously underfit and overfit the data. As C increases, the SVM will overfit the data, since it tries to classify all data correctly. On the other hand, as C decreases, it will underfit the data. An opposite behaviour applies to σ. As σ increases, it will underfit the data, whereas as σ decreases, it will

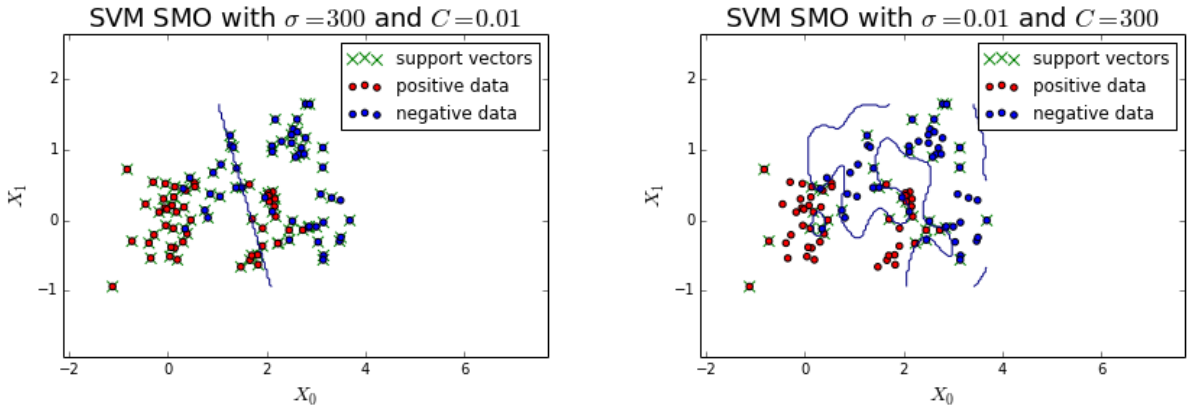
overfit the data. Figure 3a and Figure 3b show this behaviour, as we can see from the loss values that were obtained by varying C and σ, by holding the other parameter fixed. The optimal value of C, which was obtained by the Cross-Validation, was used as σ was varied and the optimal value of σ was used as C was varied. Figure 4a and Figure 4b depict the SVM SMO that underfit and overfit the data, respectively. By varying the bias parameter b, the ROC-Curve for the optimal C and σ was plotted, as shown in Figure 5



(a) Loss values with various C and $\sigma = 0.2154$

(b) Loss values with various σ and $C = 1.6681$

Figure 3: Loss values with various C and σ



(a) Underfitting with $C = 0.01$ and $\sigma = 300$

(b) Overfitting with $\sigma = 0.01$ and $C = 300$

Figure 4: Underfitting and overfitting on the *easy_2D* dataset

Assignment 5: Sklearn

Not finished yet - TODO

Assignment 6: UCI Iris Dataset

Not finished yet - TODO

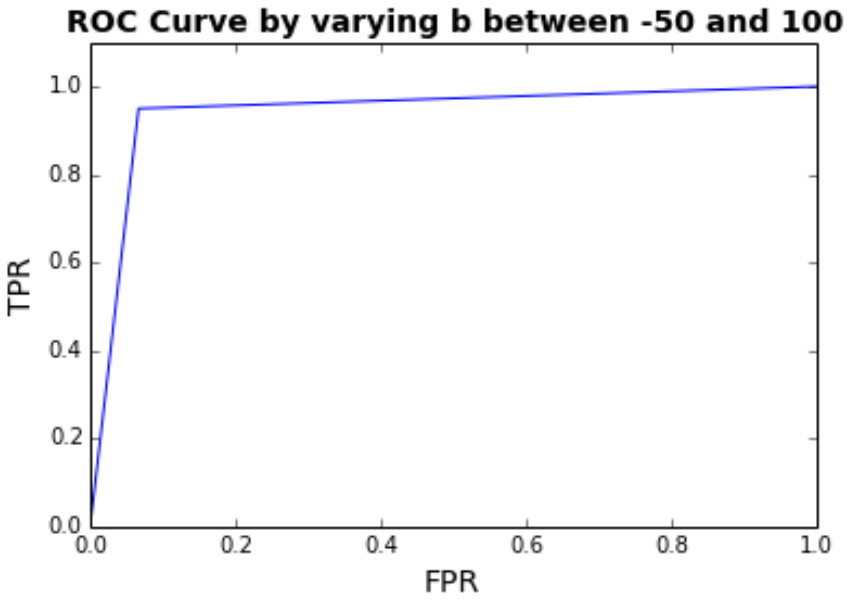


Figure 5: ROC-Curve by varying bias b between -50 and 100