Homework 6: Transductive Support Vector Machines

Christian Bock
christian.bock@bsse.ethz.ch

Dr. Michael Moor
michael.moor@bsse.ethz.ch

Leslie O'Bray
leslie.obray@bsse.ethz.ch

Prof. Dr. Karsten Borgwardt
karsten.borgwardt@bsse.ethz.ch

Submission deadline: 02.06.2021 at 14.00

Objectives

The goals of this homework are:

- to implement a variant of SVM using the **cvxopt** package.
- to implement a transductive SVM [Joachims, 1999] with label switching strategy.
- to compare inductive and transductive SVM [Joachims, 1999] on a text dataset.

Dataset

You will work on a text dataset extracted from Reuters articles compiled by Thorsten Joachims [1]. All documents are represented as feature vectors. Each feature corresponds to the tf-idf values of a word (9930 features). The dataset consists of only 10 training examples (5 positive and 5 negative) and 600 test examples. The dataset is stored in two files `train.dat` and `test.dat`. You will find the 600 test documents in `test.dat`, where the first column indicates the true class label (either 1 or -1). You will find the 10 training documents and the same 600 test documents in `train.dat`, where the first column indicates the class label (1, -1 or 0, where 0 means unlabeled test documents). The dataset is stored in sparse format. The script for reading the data is given in `run_hw6.py`.

Package

In order to run the code, the following `Python` dependencies have to be fulfilled: `numpy`, `scikitlearn`, `cvxopt`, where `cvxopt` is a python package for convex optimization. You will find the information of `cvxopt` in the following link.

`http://cvxopt.org`

[1] http://download.joachims.org/svm_light/examples/example2.tar.gz

Exercise 1

The primal problem of soft margin SVM [Cortes and Vapnik, 1995] is:

$$\begin{aligned} &\min_{w,b,\xi} \frac{1}{2}||\vec{w}||^2 + C\sum_{i=1}^{n}\xi_i \\ \text{s.t. } &y_i(\langle \vec{w}, \vec{x_i}\rangle + b) \geq 1 - \xi_i, \quad \forall i \in \{1,\dots,n\} \\ &\xi_i \geq 0, \quad \forall i \in \{1,\dots,n\} \end{aligned}$$

Its dual representation is:

$$\begin{aligned} &\max_{\alpha} W(\alpha) = \sum_{i=1}^{n}\alpha_i - \frac{1}{2}\sum_{i=1}^{n}\sum_{j=1}^{n}\alpha_i\alpha_j y_i y_j \langle \vec{x_i}, \vec{x_j}\rangle \\ \text{s.t. } &\sum_{i=1}^{n}\alpha_i y_i = 0, \\ &0 \leq \alpha_i \leq C, \quad \forall i \in \{1,\dots,n\} \end{aligned}$$

The dual representation is a quadratic programming problem:

$$\begin{aligned} &\min_{\alpha} \frac{1}{2}\alpha^T P a + q^T\alpha \\ \text{s.t. } &G\alpha \leq h, \\ &A\alpha = b \end{aligned}$$

where

$$P = y \otimes y \cdot XX^T, q = -\vec{1}, A = y^T, b = 0, G = \begin{bmatrix} -I \\ I \end{bmatrix} h = \begin{bmatrix} \vec{0} \\ C \cdot \vec{1} \end{bmatrix}$$

Given P, q, A, b, G, h, soft margin SVM can be solved using `cvxopt` package. The implementation is given in the function `train()` of file `svm_linear.py`.

Exercise 1.a The variant of SVM used in transductive SVM [Joachims, 1999] is:

$$\begin{aligned} &\min_{w,b,\xi,\xi^*} \frac{1}{2}||\vec{w}||^2 + C\sum_{i=1}^{n}\xi_i + C_-^*\sum_{j:y_j^*=-1}\xi_j^* + C_+^*\sum_{j:y_j^*=1}\xi_j^* \\ \text{s.t. } &y_i(\langle \vec{w}, \vec{x_i}\rangle + b) \geq 1 - \xi_i, \quad \forall i \in \{1,\dots,n\} \\ &y_j^*(\langle \vec{w}, \vec{x_j}\rangle + b) \geq 1 - \xi_j^*, \quad \forall j \in \{1,\dots,k\} \\ &\xi_i \geq 0, \quad \forall i \in \{1,\dots,n\} \\ &\xi_j^* \geq 0, \quad \forall j \in \{1,\dots,k\} \end{aligned}$$

Prove that its dual representation is:

$$\max_{\alpha} W(\alpha) = \sum_{i=1}^{n+k} \alpha_i - \frac{1}{2} \sum_{i=1}^{n+k} \sum_{j=1}^{n+k} \alpha_i \alpha_j y_i y_j \langle \vec{x_i}, \vec{x_j} \rangle$$

$$\begin{aligned}
\text{s.t.} \quad & \sum_{i=1}^{n+k} \alpha_i y_i = 0, \\
& 0 \le \alpha_i \le C, \quad \forall i \in \{1, \ldots, n\} \\
& 0 \le \alpha_i \le C_-^*, \quad \forall i \in \{n+1, \ldots, n+k\} \text{ and } y_i = -1 \\
& 0 \le \alpha_i \le C_+^*, \quad \forall i \in \{n+1, \ldots, n+k\} \text{ and } y_i = 1
\end{aligned}$$

Exercise 1.b Complete the implementation of the variant of SVM [Joachims, 1999] described in `Exercise 1.a` in function `train_variant()` of the file `svm_linear.py`. Perform it and the standard soft margin SVM [Cortes and Vapnik, 1995] on the given text dataset using only labeled training documents.

Exercise 2

Exercise 2.a Implement the transductive SVM algorithm [Joachims, 1999] described in Algorithm 1 in function `train()` of the file `tsvm_linear.py`. Perform it on the given dataset using both labeled and unlabeled training documents.

Algorithm 1 Transductive SVM [Joachims, 1999]

Input: X_1 and y_1, Labeled training samples
X_2, Unlabeled training samples
C_1 and C_2, Regularizers for labeled and unlabeled samples
p, Percentage of positive samples in unlabeled data X_2

Output: A weight vector $\vec{w}$ and an intercept b

$\vec{w}, b = \text{svm.train}(X_1, y_1, C_1)$
$num_+ = |X_2| * p$
The num_+ samples from X_2 with the highest value of $\vec{w} * \vec{x_j} + b$ are assigned to positive class ($y_2[j] := 1$) and the others are assigned to the negative class ($y_2[j] := -1$)
$C_2^- = 1 \times 10^{-5}, C_2^+ = (1 \times 10^{-5}) * \frac{num_+}{|X_2| - num_+}$
while $C_2^- < C_2$ or $C_2^+ < C_2$ **do**
  $\vec{w}, b, \xi_1, \xi_2 = \text{svm.train\_variant}(X_1, y_1, X_2, y_2, C_1, C_2^-, C_2^+)$
  while $\exists m, l : y_2[m] \times y_2[l] < 0$ and $\xi_2[l] > 0$ and $\xi_2[m] > 0$ and $\xi_2[m] + \xi_2[l] > 2$ **do**
    $y_2[m] = -1 \times y_2[m]$
    $y_2[l] = -1 \times y_2[l]$
    $\vec{w}, b, \xi_1, \xi_2 = \text{svm.train\_variant}(X_1, y_1, X_2, y_2, C_1, C_2^-, C_2^+)$
  $C_2^- = min(2C_2^-, C_2)$
  $C_2^+ = min(2C_2^+, C_2)$
return $\vec{w}$ and b

Exercise 2.b We did not tune the parameters of the transductive SVM [Joachims, 1999] in this homework. How would you choose regularization parameters C_1 and C_2?

Exercise 2.c To make use of the kernel trick, we map our input data $x \in \mathcal{X}$ into $\mathcal{H}$ using the map $\phi : \mathcal{X} \to \mathcal{H}$. $\mathcal{H}$ is a space of functions called the *Reproducing Kernel Hilbert Space* (RKHS). Explain why $\mathcal{H}$ called *reproducing*?

Command-line arguments

You should only complete the functions in `svm_linear.py` and `tsvm_linear.py`, and not modify the file `run_hw6.py` for submission. The file `run_hw6.py` must be executable using the following command:

```
$ python run_hw6.py
```

Note: Zero points for the **programming part** of this homework if your script is **not** executable with the above command-line argument, or the running time of your script is over **10 minutes**!

Grading and submission guidelines

This homework is worth a total of 100 points. Table 1 shows the points assigned to each exercise.

50 pts.	**Exercise 1**	
	20 pts.	Exercise 1.a
	30 pts.	Exercise 1.b
50 pts.	**Exercise 2**	
	30 pts.	Exercise 2.a
	10 pts.	Exercise 2.b
	10 pts.	Exercise 2.c

Table 1: Grading key for Homework 6

References

C. Cortes and V. Vapnik. Support-vector networks. *Machine Learning*, 20(3):273–297, 1995. doi: 10.1007/BF00994018. URL `http://dx.doi.org/10.1007/BF00994018`.

T. Joachims. Transductive inference for text classification using support vector machines. In *Proceedings of the Sixteenth International Conference on Machine Learning (ICML 1999), Bled, Slovenia, June 27 - 30, 1999*, pages 200–209, 1999.