How to Use WolfeSVM

onigiri
masashi_kitamura@mist.i.u-tokyo.ac.jp

October 23, 2014

1 Introduction

WolfeSVM is a library for solving ν-SVM [2] by using Wolfe's algorithm [3]. The algorithm of WolfeSVM is based on [1]. According [1], WolfeSVM is suited for the data whose sample size is sufficiently lager than the feature size or for the kernel funciton whose rank of kernel matrix is low.

2 Parameters

In order to use WolfeSVM, we should type command like,

"ExecuteNuSVM.exe -t "C:\SVM\DataSet\covtype" -n 0.85 -N 0.95 -v -0.01".

We can set parameters by "-key value". We explain all of the parameters we can use.

- l : This is the file path for the list of hyper-parameters ν. In the file, each line must have exactly one value. If we set this parameter, other parameters n, N, v are ignored.
- n : This is the minimum value of hyper-parameter ν. The default value is 10^{-9}.
- N : This is the maximum value of hyper-parameter ν. The default value is $1 - 10^{-9}$
- v : This is the step size of ν. Note that this value must be NEGATIVE. The default value is -0.01.

Here are some examples. If we set "-n 0.8 -N 0.9 -v -0.05" then the program solves ν-SVM with $\nu = 0.9, 0.85, 0.8$ in this order. If we set "-l nuList.txt -n 0.8 -N 0.9 -v 0.05" and if nuList.txt is as:

0.6
0.8
0.7
0.92

then the program solves ν-SVM with $\nu = 0.6, 0.8, 0.7, 0.92$ in this order. In this case, "-n 0.8 -N 0.9 -v 0.05" are ignored.

- t : This is the file path of training sample. You must set this parameter.
- T : This is the file path of test sample. If we do not want to check accuracies, we do not need to set this value.
- o : This is the file path where the program outpus the results. We do not need set this value unless you need output file.
- r : This is the results which we write in the output file. The default value is "sfntivco". We explain more details in Section 3.
- c : This is the number of core for parallelization.
- k : This value decides the kernel function.
  $\star$ 0 : linear kernel, $k(x, y) := x \cdot y$.
  $\star$ 1 : RBF kernel, $k(x, y) := \exp(-\|x-y\|^2/\gamma)$. Here, γ is a parameter.
  $\star$ 2 : polynomial kernel, $k(x, y) := (x \cdot y)^g/d$. Here, g is a parameter and d is the feature size of the dataset.

3 Outputs

If we set output file by -o option, then some results are written in that file. By -r option, we can select the results which we want to output. In this section, we will explain about this -r option.

- s : The number of training sample. The prefix is "Sapmle ".
- f : The feature size of training sample. The prefix is "Feature ".
- n : The value of hyper-parameter ν. The prefix is "Nu ".
- i : The number of iteration for finding the optimal solution. The prefix is "Iteration ".
- v : The number of support vectors. The prefix is "#SupportVector ".
- w : The coefficient vector w of classifier funciton $f(x) = w \cdot x + b$. The prefix is "W".

- b : The bias term b of classifier funciton $f(x) = w \cdot x + b$. The prefix is "B ".
- r : The margin ρ. The prefix is "Rho ".
- c : The corresponding hyper-parameter C of C-SVM in order to obtain same classifier funcion. The prefix is "C ".
- o : The optimal value. The prefix is "OptimalValue ".
- a : The accuracy. If we do not set test file, then this value is 0. The prefix is "Accuracy ".
- l : The predicting labels for the dataset in test file. The prefix is "Labels".

For example, if we set -r sfn, then output file is as below.

Sample 1257
Feature 8
Nu 0.02387

Not that W and Labels are vectors and these are outputed as below.

W
0.0765
12.8766
79.0087
Labels
1
0
0
0
1
1

au

4 Library Usage

If you are a developer then you can use WolfeSVM in C# as library. In order to use WolfeSVM, please make a instance of NuSVM class. Then call Train method. Train method returns a instance of NuSVMResult class. This has some results of experiments stated previous chapter.

References

[1] M. Kitamura, A. Takeda, and S. Iwata. Exact SVM training by Wolfe's minimum norm point algorithm. In *Proceedings of 2014 IEEE International Workshop on Machine Learning for Signal Processing*, 2014.

[2] B. Schölkopf, A. J. Smola, R. C. Williamson, and P. L. Bartlett. New support vector algorithms. *Neural Computation*, 12:1207–1245, 2000.

[3] P. Wolfe. Finding the nearest point in a polytope. *Mathematical Programming*, 11:128–149, 1976.