FLSSVM 0.1
Fortran Least Squares Support Vector Machine

José B. Colmenares

Introduction

This document is an introduction to flssvm, a program written in fortran that does the least squares version of the support vector machines developed by Vapnik [3]. As far as I am aware, the first article where the least square version appears (but not refered to with such a name) was written by Saunders *et al* [1]. This introduction will be based mostly in the work made by Suykens and others. I will cite their work when it's appropiate, but the reader should know that they have published a book [2] which even though I have not read it, promises to be very good. This introduction was born from notes I took while writting a code for LS-SVM, and later decided to write them in a clean way in the hope that it may be helpfull to someone. I expect to clean the code even more, writte it in the form of a library so it can be easily used in other programs, and improve it. It can be faster and more stable among other things, but all this takes time, and I may take some time to finish the code if you can call a code finished. Many people have shared their codes with me, written in C, matlab, scilab, and other lenguages. For instance, I'm using a laptop with ubuntu. I hope someone finds this hopefull, and helps him to discover great things to share.

A good place to start if you are looking for an introduction to support vector machines (SVM) is wikipedia. For an introduction to SVM regression a great tutorial was written by smola, (google it and you'll get many hits). For the Least Squares case, look for the paper "Weighted least squares support vector machines: robustness and sparse approximation" written by Suykens, Brabanter, Lukas and Vandewalle in Elsevier's Neurocomputing.

Installing

- Download flssvm-xx.tar.gz.
- Put the file in the directory where you want the final instalation to be.
- If using microsoft windows untar it using winrar or other similar utility. If using linux untar it using

```
tar -xvzf flssvm-xx.tar.gz
```

- Compile it. You'll need a fortran compiler and the Scons utility. put yourself in the directory where you untared the file and type

```
scons FC=your_fortran_compiler
```

if for some reason you cannot install or use scons, make a new directory called "bin" and just compile everything. In linux use:

```
your_fortran_compiler -o bin/flssvm src/flssvm_utilities.f90
  src/flssvm_predict.f90 src/flssvm_train.f90
  src/flssvm_main.f90
your_fortran_compiler -o bin/svm_r_datagen
  extra/svm_r_datagen.f90
your_fortran_compiler -o bin/svm_c_datagen
  extra/svm_c_datagen.f90
```

in microsoft windows just change the / for \.

- Add the directory where *flssvm* is to your PATH. If you are using linux how to do this depends on your distribution, but one way that usually works is to edit your .profile file at your home directory and, asuming that the directory is = /home/tferro/, add the lines

  ```
  PATH=$PATH:/home/tferro/flssvm/bin/
  export PATH
  ```

  You may need to logout and login again for this to take effect. If you are using microsoft windows go to the control panel, look for "system variables" and edit the SYSTEM PATH variable adding the equivalent of "/home/tferro/flssvm/bin/", separating it of the other directories by a semicolon.

Usage

For help on how to use flssvm, type flssvm –help. What needs to be pointed out is the format of the input and output data. The training data file is a text file. It's first line contains the number of samples n, next the dx dimension of x, and next the dimension dy of y. In the clasification case, the dy is always one so this line is omited. Next comes n lines, each containing first the dx values of the x followed the dy values of y. Two fortran programs are given as examples:

svm_r_datagen.f90 generates n points of a sinc function with random noise with an uniform distribution. This output can be use for regression training. It also generates the same number of random points to test the training. To run it do:

```
$ svm_r_datagen 100
$ flssvm k=2 s1=1.0 g=1.0
$ flssvm act=1 in=output out=result
```

in the first line a a data set of 100 points is constructed with the name "training", which is the default name for training data in flssvm. Next, the svm is trained with a RBF kernel, a kernel parameter equal to 1.0 and gamma equal to 1.0. And finally a regression is carried out. You can take a look at the results with gnuplot, a very usefull plotting utility that can be downloaded from the web, using:

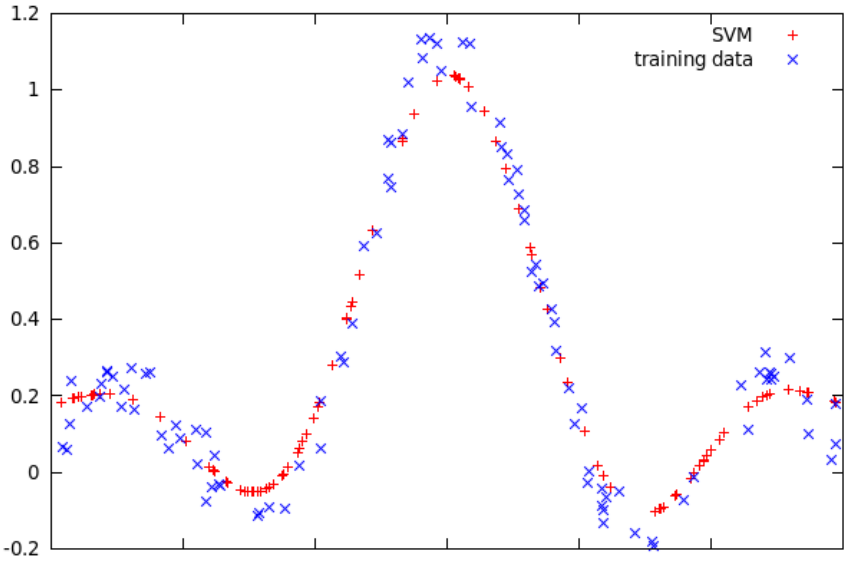

Figure 1: flssvm regression of a sinc function, with the training data having an uniform distribution noise.

```
$ gnuplot
gnuplot> set xrange[-3:3]
gnuplot> set yrange[-0.5:1.3]
gnuplot> plot 'result' with points lc rgb "red" title "SVM", \
         'training.gnuplot' with points lc rgb "blue"
         title "training data"
```

svm_c_datagen.f90 generates n points of which those inside a circle of radio 0.2 are labeled with a one, and those outside are labeled with minus one. This output can be use for clasification training. It also generates the same number of random points to test the training. To run it do (note differences with the previous example):

```
$ svm_c_datagen 1000
$ flssvm k=2 s1=0.1 g=1.0 type=0
$ flssvm act=1 in=output out=result type=0
```

in the first line a a data set of 100 points is constructed with the name "training", which is the default name for training data in flssvm. Next, the svm is trained with a RBF kernel, a kernel parameter equal to 0.1 and gamma equal to 1.0. And finally a clasification is carried out. You can take a look at the results with gnuplot using:

```
$ gnuplot
gnuplot> splot 'result' using 1:2:3
```

the image if nothing extraordinary, but you can see that the points have been clasified correctly.

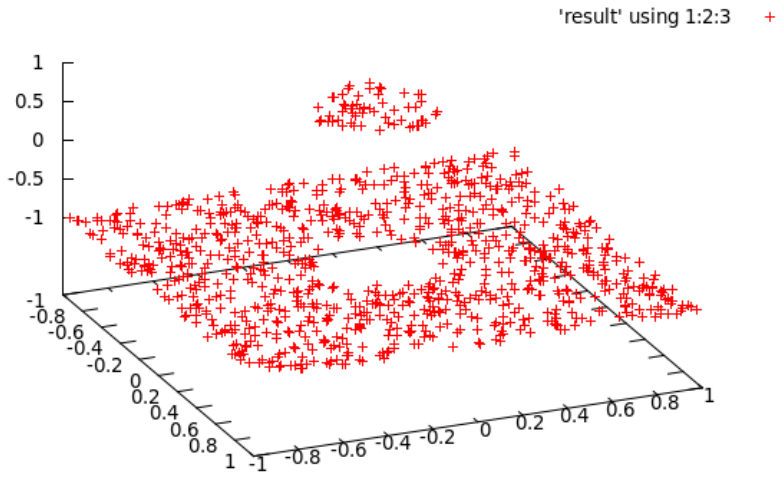

Figure 2: flssvm clasificacion.

References

[1] C. Saunders, A. Gammerman, and V. Vovk. Ridge regresion learning algorithm in dual variables. pages 512–521, 1998.

[2] J. Suykens, T. Gestel, J. Bradanter, B. Moor, and J. Vandewalle. World Scientific Publishing Co. Pte. Ltd., 2002.

[3] V. Vapnik. *The Nature of Statistical Thinking*. Springer, 1995.