CSE 455/555 Spring 2013 Homework on Discriminants and SVM

Jason J. Corso
Computer Science and Engineering
SUNY at Buffalo
jcorso@buffalo.edu

This assignment does not need to be submitted and will not be graded, but students are advised to work through the problems to ensure they understand the material.

You are both allowed and encouraged to work in groups on this and other homework assignments in this class.

Support-Vector Machines

You must use Matlab for this problem. For information on accessing or obtaining MATLAB, see `http://www.buffalo.edu/ubit/service-guides/software/purchasing/personally-owned-computers/windows/purchasing-matlab-for-windows-for-personally-owned-computers.html`. Note that MATLAB can also be accessed (for CSE students at least) by SSHing into cse servers like timberlake.cse.buffalo.edu and metallica.cse.buffalo.edu, though this will make accessing the GUI difficult unless you are running Linux.

For those not familiar with MATLAB, a set of quick introductory slides on the MATLAB interface and scripting language can be found at `http://www.cse.buffalo.edu/~jcorso/t/2013S_555/files/yingbo_matlab.pdf`.

In this problem, you will explore a straightforward implementation of a support vector machine. I have provided a skeleton set of files, which are obtainable at `http://www.cse.buffalo.edu/~jcorso/t/2013S_555/files/homework3-files.tar.gz`, in the folder labeled problem3. You should complete them, per the questions below.

Getting started. You can load a provided dataset easily:

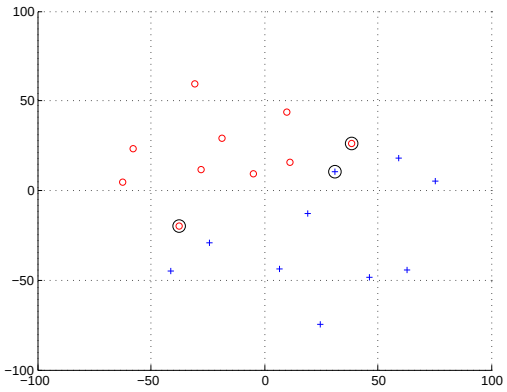
```
D = read_datafile('test1.dat'); plot_dataset(D);
```

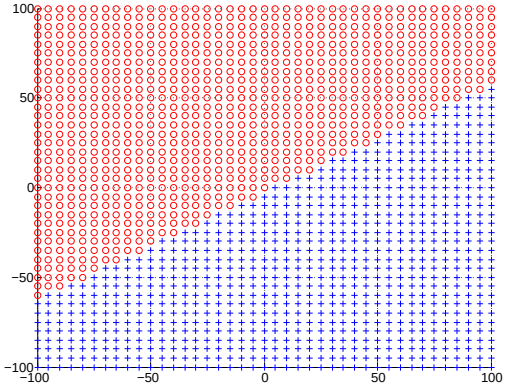
The following questions ask you to complete the SVM code by filling in the missing parts.

1. Complete line 40 in classify_grid.m, which applies a learned SVM classifier. You need this to test a classifier.
2. Complete line 48 in trainsvm.m.
3. Complete line 78 in trainsvm.m. Use the quadprog solver in matlab to actually solve SVM optimization problem.
4. Load datafile test1.dat and run

   ```
   [SV,alpha,b] = trainsvm(D,inf,@kernel_linear,[]);
   ```

   You should ultimately see figures like the ones below popping up. The function returns `SV, alpha, b`, explain these parameters.





5. Explain what the circled data points are.
6. Explain the inf in the second parameter of trainsvm.
7. Write a new kernel function that implements a quadratic kernel. Load in nonlin1.dat and run the SVM. Plot the output and explain it.
8. Load noise1.dat. Run the code with the second parameter set to inf, 10, 1, 0.1 and plot the results. What's changing and why? What role does the second parameter play?
