Project 2

Due Data: Thursday, 27 March 2014

In this project, you will code up several algorithms and try them out on simple 2D data sets. In so doing, I hope that the "magic" surrounding these algorithms will completely disappear, and that you'll have a better understanding of how and why the algorithms work as they do.

Once you understand the algorithms, I suspect that each task should only take you between 1-4 hours to implement. If you have questions, please come by to chat. Feel free to discuss with each other, but make sure you write your own code.

Tasks

You should code up the following algorithms in any language you desire, and test them out on the specified data sets.

1. Implement the perceptron algorithm. Apply it to the data sets `d1.txt` and `d2.txt`. Implement the version of the algorithm that doesn't require the decision boundary to pass through the origin.
2. Implement a linear SVM. Apply it to the data sets `d1.txt` and `d2.txt`. Again, don't require the decision boundary to pass through the origin.
3. Implement a polynomial SVM. That is, move the data to a higher-dimension feature set (quadratic) and then apply your linear SVM to the transposed data set. Apply it to data sets `d3.txt` and `d4.txt`.[1]
4. Implement bagging, with *N* "decision stumps" as the weak classifiers. Apply it to data sets `d2.txt` and `d4.txt`.
5. Implement AdaBoost, with *N* "decision stumps" as the weak classifiers. Apply it to data sets `d2.txt` and `d4.txt`.

Note: You don't need to divide the data sets into training and test sets. The data sets are the training data. Just produce the decision boundaries using each algorithm given the data (training) sets provided.

What to Turn In

You should submit:

1. The code you developed for each task. You are free to use optimization "toolboxes" when needed (such as for SVMs), but the other code should be yours.
2. A figure file showing the boundaries computed by your algorithms on each specified data set. This brief write-up can include analysis of the impact of parameters where applicable.

[1] For simplicity in plotting this decision boundary in the original feature space, I'm fine if you just classify and plot a bunch of randomly sampled vectors, which will implicitly show the decision boundary.