Supervised learning: Support Vector Machines and Random Forests

Machine Learning, CS 5333/7333

In this project you will implement and experiment with (a) the Sequential Minimal Optimization (SMO) algorithm [5] for identifying support vectors in training data and (b) a Random Forest [2] (an instance of bagging [1] ensemble learners) generator, using both artificial and real-life data sets.

1 Determining support vectors

You will implement the SMO algorithm to train Support Vector Machines (SVMs) [3]. You are required to implement and compare both the original SMO algorithm [5] (or a simplified implementation as described in the file `smo.pdf` uploaded as part of the "SVM: Background Material" item on Harvey) as well as some extension of it (for exmaple, [4]). You should select the margin parameter C using n-fold cross-validation. You are required to implement and compare both the basic SMO, which uses the inner product between the points in the training set and one that uses the Gaussian and $K(x, z) = (x^T z + c)^2$ kernels.

2 Random Forest learner

You will implement the Random Forest learning system where individual trees will be learned using the generic Decision Tree Learning algorithm presented on Slide 9 of the "Supervised Learning: Decision Trees" lecture slides posted on Harvey. For information gain measures, you should compare the performances of the following metrics: (i) misclassification, (ii) entropy, and (iii) Gini index (see Slide 16 of the lecture slides). Use the approach outlined in Slide 21 of the lecture slides to find split thresholds for real-valued features. You should show results from experiments varying the number of trees T over the set {50,100,150} and choosing M features randomly at each node, for cases where the number of features $F > 2$, where $M \in \{2, 4, \ldots, \lfloor \log_2 F + 1 \rfloor\}$.

3 Experimental domains

You are required to experiment with the following domains:

Artificial domains: The codes for generating these data sets will be provided.

- **Intertwined spiral:** The code generates two noisy intertwined spirals, where points on each of the spiral belongs to the same class.
- **k Gaussian sources:** The code generates output from k Gaussian sources in n-dimension. Vary the number of dimensions and the number of sources and compare the learning algorithms for scale-up and problem difficulty (sources closer versus further from each other).

Real-life datasets: You will use datasets from available online sources.

- **UCI ML repository datasets:** You will use at least 3 datasets from the UCI Machine Learning Repository, including the *Adult* dataset (for ease of comparison of your SMO algorithm results with those presented in Platt's paper).
- **MNIST dataset:** You will use the MNIST database of handwritten digits.

4 Report

Your report should highlight your observations from experiments to address the effects of various system parameters and the relative effectiveness of the SVM and Random Forest approaches for supervised classification for scale-up, noise, etc. The goal of the report would be to come up with some recommendation for choosing the parameters, use of kernels, training regime, etc.

Grading

The approximate breakup of grades for this project are as following:

Implementation of SMO and its improvement 30%	
Implementation of Random forest with different gain criteria	30%
Experimentation	20%
Analysis and writing style/clarity	20%
Total	100%

References

[1] Leo Breiman. Bagging predictors. *Machine Learning*, 24(2):123–140, Aug 1996.

[2] Leo Breiman. Random forests. *Machine Learning*, 45(1):5–32, Oct 2001.

[3] Marti A. Hearst, Susan T Dumais, Edgar Osuna, John Platt, and Bernhard Scholkopf. Support vector machines. *IEEE Intelligent Systems and their applications*, 13(4):18–28, 1998.

[4] S. Sathiya Keerthi, Shirish Krishnaj Shevade, Chiranjib Bhattacharyya, and Karuturi Radha Krishna Murthy. Improvements to platt's smo algorithm for svm classifier design. *Neural computation*, 13(3):637–649, 2001.

[5] John Platt. Fast training of support vector machines using sequential minimal optimization. In B. Scholkopf, C. Burges, and A. Smola, editors, *Advances in Kernel Methods: Support Vector Learning*, pages 185–208. MIT Press, Cambridge, MA, 1998.