IN4320 Machine Learning Computer Exercise

Version 0.4

February 24, 2016

Computational Learning Theory: boosting

I. Prove that $e^{-x} \geq (1 - x)$.

Before you answer the following questions, you need to read the paper 'A decision-theoretic generalization of on-line learning and an application to boosting' by Y. Freund and R.E. Schapire, 1995 (also available from Blackboard, under Course Documents, Reading Material, Computation Learning Theory). You do not have to focus too much on sections 1, 2 and 3, but section 4 and 4.1 are important.

a. Implement a 'weak learner': the decision stump. The decision stump is a very simple classifier that chooses one feature f from a dataset, and checks if the feature value x_f is larger (or smaller) than a certain threshold θ. If $x_f > (<)\theta$ then the object is assigned to class ω_1 and otherwise to ω_2.

To find the optimal feature f and threshold θ, you have to do an exhaustive search for a training set. So, try all values for f and θ and the sign y of $<$ or $>$, and remember those values f^*, θ^*, y^* for which the classification error on the trainingset is minimum.

Make sure that the function accepts a dataset with labels as input, and that the function outputs the optimal f^*, θ^* and y^*.

b. Test the implementation on the dataset generated by `gendats` from Prtools. (If you don't want to use Prtools, just generate the two classes from two Gaussian distributions, where the means are $\mu_1 = [0, 0]^T$ and $\mu_2 = [2, 0]^T$, and the covariance matrices are identify matrices.) Make a scatterplot of the data, and give the optimal parameters obtained by your decision stump.

c. Test the implementation on dataset `optdigitsubset` of last week. Use the first 50 objects for each class for training, and the rest for testing. What is the classification error on the test objects? How much does this vary when you take other random subsets of 50 for training?

d. Extend the implementation of the weak learner such that it accepts a weight per object. Therefore, next to a dataset and labels, the function should accept a weight $w_i > 0$ per object $\mathbf{x}_i$. The weighted weak learner should now minimize the weighted classification error.

Test the code by using a simple classification problem again, and convince yourself that the code is Bug Free.

e. Implement the algorithm that is described in Figure 2 of the paper: AdaBoost.

f. Test your implementation on some dataset. Not only use a simple dataset like `gendats` but also a more complicated dataset like `gendatb`. Find out which objects obtain a large weight w_i^t. Keep the number of iterations T fixed to, say, $T = 100$.

g. Test the implementation on dataset `optdigitsubset` of last week. Use the first 50 objects for each class for training, and the rest for testing. What is the classification error on the test objects? How does this error depend on the number of iterations T? If you take the classifier with the optimal T, which training objects get a high weight? Visualize that in a scatter plot.