CSI 5325 Assignment 3

Greg Hamerly

Assigned: 3/23/2015; Due: 4/9/2015

Instructions

For this assignment, you should write up your derivations carefully, clearly, and concisely in LaTeX. Make them easy to follow, with appropriate English descriptions rather than just mathematics (i.e. explain what you are doing as necessary).

Write any programs in Matlab (or Octave), and attach their code with your assignment. Include any relevant figures and graphs (plotted using Matlab / Octave) in your writeup.

Submit your assignment in two ways: hardcopy (in class) and by email (to hamerly@cs.baylor.edu). The email should contain a single attachment as a ZIP file. It should be named "lastname-xx.zip", where lastname is your last name, and xx is the number of the assignment.

Finally, please keep your submitted email attachments small. In particular, make sure you are only submitting things that are necessary (omit datasets I gave you, compiled programs, etc.). Also, try to keep your graphics small by using vector (rather than bitmap) formats (e.g. PDF or EPS rather than JPG or BMP). Vector graphics are generally smaller in size and better quality than bitmap.

1 Uniform convergence (20 points)

In class we proved that for any finite set of hypotheses $H = \{h_1, \ldots, h_k\}$, if we pick the hypothesis $\hat{h}$ that minimizes the training error on a set of m examples, then with probability at least $(1 - \delta)$,

$$\epsilon(\hat{h}) \leq \left(\min_i \epsilon(h_i)\right) + 2\sqrt{\frac{1}{2m} \log \frac{2k}{\delta}},$$

where $\epsilon(h_i)$ is the generalization error of hypothesis h_i. Now consider a special case (often called the *realizable case*) where we know, a priori, that there is some hypothesis in our class H that achieves zero error on the distribution from which the data is drawn. Then we could obviously just use the above bound with $\min_i \epsilon(h_i) = 0$; however, we can prove a better bound than this.

(a) Consider a learning algorithm which, after looking at m training examples, chooses some hypothesis $\hat{h} \in H$ that makes zero mistakes on this training data. (By our assumption, there is at least one such hypothesis, possibly more.) Of course, this process may yield a hypothesis that makes no mistakes in training, but still has non-zero generalization error. Show that with probability $(1 - \delta)$

$$\epsilon(\hat{h}) \leq \frac{1}{m} \log \frac{k}{\delta}.$$

That is, the generalization error of the chosen hypothesis is bounded above by the given quantity. Notice that since we do not have a square root here, this bound is much tighter. [Hint: Consider the probability that a hypothesis with generalization error greater than γ makes no mistakes on the training data. Instead of the Hoeffding bound, you might also find the following inequality useful: $(1-\gamma)^m \leq e^{-\gamma m}$.]

(b) Rewrite the above bound as a sample complexity bound, i.e., in the form: for fixed δ and γ, for $\epsilon(\hat{h}) \leq \gamma$ to hold with probability at least $(1-\delta)$, it suffices that $m \geq f(k, \gamma, \delta)$ (i.e., $f(\cdot)$ is some function of k, γ, and δ).

2 Mistake bounds (20 points)

Consider learning by selecting a hypothesis $h : \mathcal{X} \to \{0, 1\}$ from a finite class H. Suppose we make two rather strong assumptions:

- We assume that all our training data were correct (i.e. there is no noise). Thus, during training, if a hypothesis $h \in H$ ever misclassifies an example, we know h must be wrong.
- We assume that the correct hypothesis exists in H.

Consider now the following algorithm which observes a sequence of examples from a sample S, one at a time. For each example, it makes a prediction using a majority vote of the remaining hypotheses. It then discards incorrect hypotheses based on their individual correctness. Note that this algorithm makes predictions 'in real time', one example at a time, rather than examining all the examples first.

Algorithm 1 HALVING(H, S)

```
numMistakes ← 0
for all (x, y) ∈ S do
    s ← Σ_{h∈H} h(x)
    p ← 1{s ≥ |H|/2} {the prediction}
    numMistakes ← numMistakes + 1{p ≠ y}
    remove from H each h ∈ H where h(x) ≠ y {remove incorrect hypotheses}
end for
return (H, numMistakes)
```

Hopefully, when this algorithm finishes, we have eliminated all hypotheses but the correct one. In class we have discussed bounds on sample complexity and error. In this setting, we are interested in the number of mistakes that the HALVING algorithm makes (i.e. the number of incorrect predictions made by the majority vote).

(a) Prove that the following inequality holds:

$$\text{numMistakes} \leq \log_2(|H|).$$

[Hint: think about how many incorrect hypotheses are eliminated at each round of the algorithm, and how it relates to the correctness of the majority prediction for that round.]

(b) What is the least number of mistakes that the algorithm might make before getting down to $|H| = 1$?

3 Experiments with uniform convergence and mistake bounds (20 points)

Define your own simple learning problem, like a 2-class classification task using real-valued features. Make your own dataset, with both training and test data (at *least* 100 examples each, though many more are fine). Define a small, finite class of hypotheses H (as we assume in our discussions of uniform convergence and mistake bounds). Use at least 10 hypotheses in H, though many more are fine.

Then run some experiments to test how well the theoretical bounds work. Here are some questions to consider (you should answer these, but also consider how you can go further):

- How good is the estimator $\epsilon(\hat{h})$? You may need to do multiple tests with multiple train/test datasets, so having a program to generate datasets would be helpful.
- How good are the mistake bounds? Again, using multiple train/test datasets would be a good thing to get a robust picture.

Note that using a finite class of hypotheses means that you can use the particular model

$$h(x) = (0.5 + 0.1x) > 3,$$

(i.e. that fixed function is one hypothesis), but you cannot use a model where real-valued parameters are unrestricted and learned from the data (because that makes your hypothesis class infinitely large). So here, 'training' really refers to 'choose the hypothesis $\hat{h} \in H$ with the least error on the training data'.

Plots and/or tables would be good ways to explain your findings for this question. Include your source code in the electronic submission.

4 ℓ_2 norm soft margin SVMs (20 points)

In class, we saw that if our data is not linearly separable, then we need to modify our support vector machine algorithm by introducing an error margin that must be minimized. Specifically, the formulation we have looked at is known as the ℓ_1 norm soft margin SVM. In this problem we will consider an alternative method, known as the ℓ_2 norm soft margin SVM. This new algorithm is given by the following optimization problem (notice that the slack penalties are now squared):

$$\begin{aligned} \min_{w,b,\xi} \quad & \frac{1}{2}||w||^2 + \frac{C}{2}\sum_{i=1}^{m}\xi_i^2 \\ s.t. \quad & y^{(i)}(w^T x^{(i)} + b) \geq 1 - \xi_i, \quad i = 1, \ldots, m \end{aligned}$$

(a) Notice that we have dropped the $\xi_i \geq 0$ constraint in the ℓ_2 problem. Show that these non-negativity constraints can be removed. That is, show that the optimal value of the objective will be the same whether or not these constraints are present.

(b) What is the Lagrangian of the ℓ_2 soft margin SVM optimization problem?

(c) Find the critical (minimal) points of the Lagrangian with respect to w, b, and ξ by taking the following gradients: $\nabla_w \mathcal{L}$, $\frac{\partial \mathcal{L}}{\partial b}$, and $\nabla_\xi \mathcal{L}$, and setting them equal to 0. Here $\xi = [\xi_1, \ldots, \xi_m]^T$.

(d) What is the dual of the ℓ_2 soft margin SVM optimization problem?

5 VC dimension (20 points)

Let the input domain of a learning problem be $\mathcal{X} = \mathbb{R}$. Give the VC dimension for each of the following classes of hypotheses. In each case, if you claim that the VC dimension is d, then you need to show that the hypothesis class can shatter d points, and explain why there are no sets of $d+1$ points that it can shatter.

- $h(x) = 1\{a < x\}$, with parameter $a \in \mathbb{R}$.
- $h(x) = 1\{a < x < b\}$, with parameters $a, b \in \mathbb{R}$.
- $h(x) = 1\{a \sin(x) > 0\}$, with parameter $a \in \mathbb{R}$.
- $h(x) = 1\{\sin(x + a) > 0\}$, with parameter $a \in \mathbb{R}$.