CS 5350/6350: Machine Learining Spring 2015

Homework 4 Solution

Handed out: Mar 9, 2015
Due date: Mar 30, 2015

1 Margins

The margin of a set of points $\{\mathbf{x}_1, \mathbf{x}_2, \cdots, \mathbf{x}_m\}$ with respect to a hyperplane is defined as the distance of the closest point to the hyperplane $\mathbf{w} \cdot \mathbf{x} = \theta$. The margin γ is thus:

$$\gamma = \min_i \left| \frac{\mathbf{w} \cdot \mathbf{x}_i - \theta}{||\mathbf{w}||} \right|$$

Suppose our examples are points in $\{0,1\}^{20}$ (that is, 20 dimensional binary vectors). For all the questions below, we wish to learn the following concept in this space using examples:

$$f(\mathbf{x}) = \mathbf{x}_2 \vee \mathbf{x}_4 \vee \mathbf{x}_6 \vee \mathbf{x}_{10} \vee \mathbf{x}_{12} \vee \mathbf{x}_{14} \vee \mathbf{x}_{16} \vee \mathbf{x}_{18}.$$

The variables that are in the disjunction are referred to as relevant variables and the others as irrelevant.

1. [3 points] Represent f as a linear threshold function. That is, find $\mathbf{w}$ and θ such that $sgn(\mathbf{w}^T\mathbf{x} - \theta)$ is equivalent to f for all $\mathbf{x} \in \{0,1\}^{20}$.

> **Solution:**
>
> $\mathbf{w} = [2:1, 4:1, 6:1, 10:1, 12:1, 14:1, 16:1, 18:1]$
>
> $\theta = 0.5$ (any $0 < \theta < 1$ is correct, but will yield different results for the other parts of the problem)

2. [6 points] Consider a dataset D_1 that is generated as follows:

   - Positive examples: All possible points that have **one relevant variable** and **six irrelevant variables** set to *one* and all others *zero*.
   - Negative examples: All possible points that have **no relevant variables** and **six irrelevant variables** set to *one* and all others *zero*.

Compute the margin of D_1 with respect to your $\mathbf{w}$.

Solution:

$$\gamma = \min\left(\left|\frac{1-\theta}{\sqrt{8}}\right|, \left|\frac{0-\theta}{\sqrt{8}}\right|\right) = \frac{0.5}{\sqrt{8}}$$

3. [6 points] Consider a dataset D_2 that is similar to D_1, except that for positive examples **six relevant variables** are set to one in addition to the six irrelevant variables. Compute the margin of D_2 with respect to your $\mathbf{w}$.

Solution:

$$\gamma = \min\left(\left|\frac{6-\theta}{\sqrt{8}}\right|, \left|\frac{0-\theta}{\sqrt{8}}\right|\right) = \frac{0.5}{\sqrt{8}}$$

4. [6 points] Now, consider a dataset D_3 that is similar to D_1. The only difference is that the number of irrelevant variables that are seen in both positive and negative examples is increased to **ten**. Write the Perceptron mistake bound for D_1, D_2 and D_3.

Solution:

$$\gamma = \min\left(\left|\frac{1-\theta}{\sqrt{8}}\right|, \left|\frac{0-\theta}{\sqrt{8}}\right|\right) = \frac{0.5}{\sqrt{8}}$$

The mistake bound is R^2/γ^2 where R is the magnitude of the largest example. For all data sets $\gamma^2 = 1/32$.

Data Set	R^2	Mistake Bound
D_1	7	224
D_2	12	384
D_3	11	352

5. [4 points] Rank the datasets in terms of "ease of learning". Justify your answer.

Solution:

In order of easiest to hardest: D_1, D_3, D_2. The easiest to learn is the dataset the algorithm needs to make the fewest mistakes on to learn the function.

2 Boosting and Perceptron

Consider the following set of training examples (i is the index of the example):

i	x	y	label
1	1	10	-
2	4	4	-
3	8	7	+
4	5	6	-
5	3	16	-
6	7	7	+
7	10	14	+
8	4	2	-
9	4	10	+
10	8	8	-

In this problem you will use two learning algorithms, Boosting and Perceptron to learn a hidden Boolean function from this set of examples.

1. [10 points] Use two rounds of AdaBoost to learn a hypothesis for this Boolean data set. As weak learners use a hypothesis of the form $[x > \theta_x]$ or $[y > \theta_y]$, for some integer θ_x or θ_y. Each round, choose the weak learner that minimizes the error ϵ. There should be no need to try many θs, appropriate values should be clear from the data.

   Start the first round with a uniform distribution D_0. Place the value for D_0 for each example in the appropriate column of the table. Find the hypothesis given by the weak learner that minimizes the error ϵ for that distribution. Place this hypothesis as the heading to the fourth column of the table, and give its prediction for each example in that column.

   Now compute D_1 for each example, and select hypothesis that minimizes error on this distribution, placing these values and predictions in the appropriate columns of the table.

Solution:

For this problem the error term is defined as

$$\epsilon_t = \frac{1}{2} - \frac{1}{2}\sum_{i=1}^{10} D_t(i)\, y_i\, h_t(x_i)$$

where h_t is the weak hypothesis for this iteration. The weight of the iteration is defined as

$$\alpha_t = \frac{1}{2}\ln\left(\frac{1-\epsilon_t}{\epsilon_t}\right)$$

and the weight for the data during the next iteration is given by

$$D_{t+1}(i) = \frac{D_t(i)}{Z_t} \exp(-\alpha_t \, y_i \, h_t(x_i))$$

where Z_t is a normalizing constant such that $\sum D_{t+1} = 1$.

For the first iteration:

$$\epsilon_0 = \frac{1}{2} - \frac{1}{2} \sum_{i=1}^{10} D_0(i) \, y_i \, h_0(x_i) = 0.2$$

$$\alpha_0 = \frac{1}{2} \ln \left(\frac{1 - \epsilon_0}{\epsilon_0} \right) = 0.693$$

Given all $D_0(i) = 0.1$ and the value of α_0, we find $Z_0 \, D_1(i)$ is either 0.05 for a correct classification or 0.2 for an incorrect classification. h_0 misclassifies 2 examples, so $Z_0 = 0.8$. This gives the following values for ϵ_1 and α_1.

$$\epsilon_1 = \frac{1}{2} - \frac{1}{2} \sum_{i=1}^{10} D_1(i) \, y_i \, h_1(x_i) = 0.25$$

$$\alpha_1 = \frac{1}{2} \ln \left(\frac{1 - \epsilon_1}{\epsilon_1} \right) = 0.549$$

		Boosting				**Perceptron updates**
		Hypothesis 1		Hypothesis 2		
i	Label	D_0	$\theta_x = 6$	D_1	$\theta_y = 9$	Start with $[0, 0, -3]$
1	-	0.1	-	0.0625	+	
2	-	0.1	-	0.0625	-	
3	+	0.1	+	0.0625	-	
4	-	0.1	-	0.0625	-	
5	-	0.1	-	0.0625	+	
6	+	0.1	+	0.0625	-	
7	+	0.1	+	0.0625	+	
8	-	0.1	-	0.0625	-	
9	+	0.1	-	0.25	+	
10	-	0.1	+	0.25	-	

2. [5 points] Write the final hypothesis produced by AdaBoost.

Solution:

The class predicted by AdaBoost is the class with the highest weight from all the weak hypotheses. For the binary case this can be written as

$$H(x) = \text{sgn}\left(\sum_i \alpha_i h_i(x)\right)$$

for this problem there are only two iterations, so

$$H(x) = \text{sgn}(0.693\, h_0(x) + 0.549\, h_1(x))$$

3. [6 points] Use the Perceptron learning algorithm to train a hypothesis for this Boolean data set, using as features the weak learners chosen by the boosting algorithm in (a).

   Train the Perceptron one cycle through the data, using 0 as initial weights, threshold of 3, and learning rate of 1. Fill in the *weight vector* column of the table.

Solution:

There was some confusion about the threshold of 3. In the literature this is used in the context $\mathbf{w}^T\mathbf{x} > b$ where b is the "threshold." For this problem I start with $w_0 = [0, 0, -3]$ and all inputs $x' = [h_0(x), h_1(x), 1]$, but other sane solutions should not lose points. The perceptron update in this context is

$$w_{i+1} = w_i + yx$$

if there is a mistake ($y_i \mathbf{w}^T \mathbf{x}_i \leq 0$).

		Boosting				Perceptron updates
		Hypothesis 1		Hypothesis 2		
i	Label	D_0	$\theta_x = 6$	D_1	$\theta_y = 9$	Start with $[0, 0, -3]$
1	-	0.1	-	0.0625	+	$[0, 0, -3]$
2	-	0.1	-	0.0625	-	$[0, 0, -3]$
3	+	0.1	+	0.0625	-	$[1, -1, -2]$
4	-	0.1	-	0.0625	-	$[1, -1, -2]$
5	-	0.1	-	0.0625	+	$[1, -1, -2]$
6	+	0.1	+	0.0625	-	$[2, -2, -1]$
7	+	0.1	+	0.0625	+	$[3, -1, 0]$
8	-	0.1	-	0.0625	-	$[3, -1, 0]$
9	+	0.1	-	0.25	+	$[2, 0, 1]$
10	-	0.1	+	0.25	-	$[1, 1, 0]$

4. [4 points] Did the two algorithms converge to the same hypothesis? If both hypotheses were used to predict labels for the training set, would the set of predictions be the same? Explain.

Solution:

AdaBoost relies on h_0 if there is a disagreement, so H exactly matches h_0 producing misclassifications for the last two examples.

Perceptron gives $\mathbf{w}^T\mathbf{x} = 0$ whenever there is a disagreement between h_0 and h_1. Depending on your interpretation of sgn(0) this produces 2, 3, or 5 misclassifications, none of which are the same as AdaBoost.

3 Kernels

(a) [10 points] If $K_1(\mathbf{x}, \mathbf{z})$ and $K_2(\mathbf{x}, \mathbf{z})$ are both valid kernel functions, with positive α and β, prove that

$$K(\mathbf{x}, \mathbf{z}) = \alpha K_1(\mathbf{x}, \mathbf{z}) + \beta K_2(\mathbf{x}, \mathbf{z}) \tag{1}$$

is also a kernel function.

Solution:

From the lecture slides:

(1) If K is a valid kernel then aK is also a valid kernel.

(2) If K_1 and K_2 are both valid kernels then $K_1 + K_2$ is a valid kernel.

The first rule shows αK_1 and βK_2 are both valid kernels. The second rule shows $\alpha K_1 + \beta K_2$ is a valid kernel.

(b) [10 points] Given two examples $\mathbf{x} \in \Re^2$ and $\mathbf{z} \in \Re^2$, let

$$K(\mathbf{x}, \mathbf{z}) = (\mathbf{x}^T\mathbf{z})^3 + 9(\mathbf{x}^T\mathbf{z})^2 + 81\mathbf{x}^T\mathbf{z}. \tag{2}$$

Prove that this is a valid kernel function.

Solution:

From the lecture slides:

(3) If K is a valid kernel then K^n is also a valid kernel.

$\mathbf{x}^T\mathbf{z}$ is the linear kernel, and powers of this kernel are also valid kernels. The rules from part (a) show linear combinations of kernels are also kernels, so $K(\mathbf{x}, \mathbf{z})$ is a kernel.

4 Learning Decision Lists (For CS 6350 students)

In this problem, we are going to learn the class of k-decision lists. A decision list is an ordered sequence of if-then-else statements. The sequence of if-then-else conditions are tested in order, and the answer associated to the first satisfied condition is output (see Figure 1).

A *k-decision list* over the variables $x_1, \ldots, x_n$ is an ordered sequence $L = (c_1, b_1), \ldots, (c_l, b_l)$ and a bit b, in which each c_i is a conjunction of at most k literals over $x_1, \ldots, x_n$. The bit b is called the *default* value, and b_i is referred to as the bit *associated* with condition c_i. For any input $x \in \{0,1\}^n$, $L(x)$ is defined to be the bit b_j, where j is the smallest index satisfying $c_j(x) = 1$; if no such index exists, then $L(x) = b$.

We denote by *k-DL* the class of concepts that can be represented by a k-decision list.

1. [8 points] Show that if a concept c can be represented as a k-decision list so can its complement, $\neg c$. You can show this by providing a k-decision list that represents $\neg c$, given $c = \{(c_1, b_1), \ldots, (c_l, b_l), b)$.

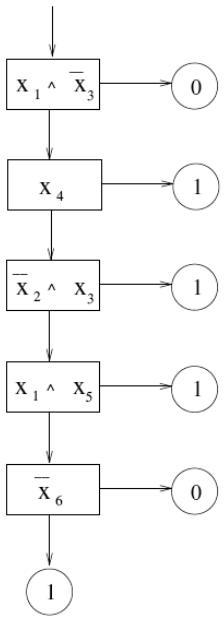

Figure 1: A 2-decision list.

Solution:

Inverting all the outputs is equivalent to inverting the list.

$$\neg c = \{(c_1, \neg b_1), \ldots, (c_t, \neg b_t), \neg b\}$$

2. [9 points] Use Occam's Razor to show:
For any constant $k \geq 1$, the class of k-decision lists is PAC-learnable.

Solution:

To show k-decision lists are PAC learnable we will show $\log(|H|)$ is bounded by a polynomial. The number of conjunctions with at most k terms using n variables is $\sum_{i=1}^{k} \binom{n}{i}$. This has an upper bound of n^k. Each decision has two possible leaves and can be ordered any way. The number of k-decision lists is then $O(n^k!)$. Since $\log n!$ has an upper bound of $n \log n$, $\log(|H|)$ is $O(n^k \log n^k)$, thus k-decision lists are PAC learnable.

3. [8 points] Show that 1-decision lists are a linearly separable functions. (Hint: Find a weight vector that will make the same predictions a given 1-decision list.)

Solution:

This is a 1-decision list so each decision will have only one variable. Let n be the decision index with the first decision having $n = 0$. Assume the leaf nodes are $b_n \in \{1, -1\}$ and there are N total decisions. If a variable x_i appears in the list then $w_i = b_n \, 2^{-n}$ otherwise $w_i = 0$. The bias term is $b_N \, 2^{-N}$.