Homework 4: Adaboost & Multi-class - CM146 Winter 2018*

March 13, 2018

i	Label	Hypothesis 1				Hypothesis 2			
		D_0	$f_1 \equiv [x > 2]$	$f_2 \equiv [y > 5]$	$h_1 \equiv [f_1]$	D_1	$f_1 \equiv [x > 10]$	$f_2 \equiv [y > 11]$	$h_2 \equiv [f_2]$
(1)	(2)	(3)	(4)	(5)	(6)	(7)	(8)	(9)	(10)
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	+	+	+	0.25	-	-	-
7	+	0.1	+	+	+	0.0625	+	-	-
8	-	0.1	-	-	-	0.0625	-	-	-
9	+	0.1	-	+	-	0.25	-	+	+
10	+	0.1	+	+	+	0.0625	-	-	-

Table 1: Table for Boosting results

1. (a) We note that $D_0(i) = 0.1$ for all ten examples. The best learner aligned to the y-axis is $f_1 \equiv [x > \{2\}]$ and the best learner aligned to the x-axis is $f_2 \equiv [y > \{5\}]$. We choose the former as h_1 since

$$\epsilon_{x1} = [\text{weighted sum of mistakes if } h = x_1] = \frac{2}{10}$$

$$\epsilon_{x2} = [\text{weighted sum of mistakes if } h = x_2] = \frac{3}{10}$$

*Material taken from CS446 Illinois

If base 2 is used, we get the following:

$$\alpha_0 = \frac{1}{2}\log_2\frac{1-\epsilon}{\epsilon} = \frac{1}{2}\log_2\frac{0.8}{0.2} = 1$$

If base e is used, we instead get:

$$\alpha_0 = \frac{1}{2}\ln\frac{1-\epsilon}{\epsilon} = \frac{1}{2}\ln\frac{0.8}{0.2} = \ln 2 \approx 0.693$$

(b) Using $\alpha_0(base2)$ to compute the new distribution, we get:

$$D_{t+1}(i) = \begin{cases} \frac{1}{Z_0}D_0(i)2^{-\alpha_t} & \text{if } h_t(x_i) = y_i \\ \frac{1}{Z_0}D_0(i)2^{\alpha_t} & \text{if } h_(x_i) \neq y_i \end{cases}$$

$$D_1(i) = \begin{cases} \frac{1}{20Z_0} & \text{if } h_t(x_i) = y_i \\ \frac{1}{5Z_0} & \text{if } h_t(x_i) \neq y_i \end{cases}$$

To calculate Z_0,

$$\frac{8}{20Z_0} + \frac{2}{5Z_0} = 1 \Rightarrow Z_0 = \frac{4}{5}$$

The new distribution D_1 is thus

$$D_1(i) = \begin{cases} 0.0625 & \text{if } h_1(x_i) = y_i \\ 0.25 & \text{if } h_i(x_i) \neq y_i \end{cases}$$

(Note that we get the same final result if base e is used instead)

To choose the next candidate functions we simply observe the data and choose $f_1 \equiv [x > \{10\}]$ and the best learner aligned to the x-axis is $f_2 \equiv [y > \{11\}]$.

$$\epsilon_{f1} = [\text{weighted sum of mistakes if } h = f_1] = 1 \times 0.25 + 2 \times 0.0625 = 0.385$$

$$\epsilon_{f2} = [\text{weighted sum of mistakes if } h = f_2] = 0 \times 0.25 + 4 \times 0.0625 = 0.25$$

Clearly f_2 is the winner. Then we have:

$$\alpha_1 = \frac{1}{2}\log_2\frac{1-\epsilon_{f2}}{\epsilon_{f2}} = \frac{1}{2}\log_2\frac{0.75}{0.25} = 0.79$$

or, using base e:

$$\alpha_1 = \frac{1}{2}\ln\frac{1-\epsilon_{f2}}{\epsilon_{f2}} = \frac{1}{2}\ln\frac{0.75}{0.25} \approx 0.54$$

(c) See the answer to (b).

(d)

$$H(x) = sgn(1 \times [x > 2] + 0.79 \times [y > 11])$$

If we use natural logarithms, the final hypothesis is just a scaled equivalent:

$$H(x) = sgn(0.69 \times [x > 2] + 0.54 \times [y > 11])$$

Note: a correct solutions to this problem should use the same base for all the calculations in this problem.

2. (30 points) [Multi-class classification]

(a) i. Number of classifiers

- For the **One vs. All** scheme, we will have k classifiers.
- For the **All vs. All** scheme, we will have $\binom{k}{2} = \frac{k(k-1)}{2}$ classifiers.

ii. Number of examples used to learn each classifier

- Each classifier in **One vs. All** scheme is learned over m examples $-\frac{m}{k}$ "positive" and $(m - \frac{m}{k}) = \frac{(k-1)m}{k}$ "negative."
- In the **All vs. All** scheme, each classifier is learned over only $\frac{2m}{k}$ examples $-\frac{m}{k}$ "positives" and another $\frac{m}{k}$ "negative".

iii. Labeling a new example x

- For the **One vs. All** scheme, assume that the k classifiers correspond to k weight vectors, w_1, w_2, ... w_k, and each classifier classifies an example x as positive or negative based on $sgn(w_i \ldots x \geq 0)$. In general though, we can choose the label that achieves the highest score, i.e. $y* = \arg\max w_i \ldots x$.
- For the **All vs. All** scheme, we have several options.
  One approach would be to apply all the $\binom{k}{2}$ classifiers on example x and let each classifier vote on the class label. The label with highest number of votes would be the winner.
  Another approach is to conduct a tournament between the labels.

iv. Computational complexity

- For the **One vs. All** scheme, we have k classifiers, each learned over m examples. So, the computational complexity is $O(mk)$.
- For the **All vs. All** scheme, we have $\binom{k}{2}$ classifiers, each learned over $\frac{2m}{k}$ examples. So, the computational complexity is $O(\frac{2m}{k} \times \frac{k(k-1)}{2}) = O(mk)$.

(b) Based on the analysis above, we see that both schemes are of the same order of complexity, O(mk). So, either scheme is fine, when using simple Perceptron-style classifier.

(c) Recall that the KERNEL PERCEPTRON has the computational complexity of $O(m^2)$, where m is the number of examples used by the training algorithm. Note that this is different from the simple Perceptron (which is of order $O(m)$.
This changes the analysis we did earlier.

- For the **One vs. All** scheme, we have k classifiers, each learned over m examples. So, the computational complexity of using KERNEL PERCEPTRON is $O(m^2k)$.
- For the **All vs. All** scheme, we have $\binom{k}{2}$ classifiers, each learned over $\frac{2m}{k}$ examples. So, the computational complexity of using KERNEL PERCEPTRON is $O(\frac{4m^2}{k^2} \times \frac{k(k-1)}{2} = O(m^2)$.

So, when using KERNEL PERCEPTRON, we would prefer the **All vs. All** scheme over the **One vs. All** scheme.

(d)

- For the **One vs. All** scheme, we have k classifiers, each learned over m examples. So, the computational complexity of using the blackbox learning algorithm is $O(m^2kd)$.

- For the **All vs. All** scheme, we have $\binom{k}{2}$ classifiers, each learned over $\frac{2m}{k}$ examples. So, the computational complexity of using blackbox learning algorithm is $O(d\frac{4m^2}{k^2} \times \frac{k(k-1)}{2} = O(m^2 d)$.

(e)
- For the **One vs. All** scheme, we have k classifiers, each learned over $\frac{2m}{k}$ examples. So, the computational complexity of using blackbox learning algorithm is $O(m^2 kd)$.
- For the **All vs. All** scheme, we have $\binom{k}{2}$ classifiers, each learned over $\frac{2m}{k}$ examples. So, the computational complexity of using blackbox learning algorithm is $O(d^2\frac{4m}{k} \times \frac{k(k-1)}{2} = O(d^2 mk)$.

(f)
- For the **Counting** scheme, we need to run each classifier once on the given example, which is $\frac{m(m-1)}{2}$. Time complexity is $O(m^2)$.
- For the **Knockout** scheme, each time, we will eliminate one class, and in order to pick the final winner, we need to run $(m-1)$ classifier. Time complexity is $O(m)$.