CS 5350/6350: Machine Learining Fall 2017

Homework 2
uNID: u1143683

September 26, 2017

1 Warm up: Linear Classifiers and Boolean Functions

1. $x_1 \vee \neg x_2 \vee x_3$
$$x_1 - x_2 + x_2 \geq 0$$
2. $(x_1 \vee x_2) \wedge (x_2 \vee x_3)$
$$x_1 + 2x_2 + x_3 \geq 2$$
3. $(x_1 \wedge \neg x_2) \vee x_3$
$$x_1 - x_2 + 2x_3 \geq 1$$
4. x_1 xor x_2 xor x_3
This function is not linearly seperable
5. $\neg x_1 \wedge x_2 \wedge \neg x_3$
$$-x_1 + x_2 - x_3 \geq 1$$

2 Mistake Bound Model of Learning

1.

   (a) Since $1 \leq l \leq 80$, the size of concept class $|C| = 80$

   (b) The prediciton made by hypothesis of length l on inputs x_1^t and x_2^t can be represented by the equation
$$sgn(2l - |x_1^t| - |x_2^t|)$$
   We make a mistake when this prediction does not match $sgn(y^t)$ of the label y^t. Hence we can determine whether mistake has been made using the following inequality:
$$\boxed{y^t(2l - |x_1^t| - |x_2^t|) < 0} \tag{1}$$

(c) Let us assume $l = l_c$ for correct function and $l = l_h$ for current hypothesis function. Let us consider what happens during positive and negative examples:

i. Positive Example:
When there is a mistake on positive example, we have label $y^t = +1$ and equation $2l - |x_1^t| - |x_2^t| < 0$. Therefore we can say that either x_1^t, x_2^t or both were greater than l_h and we correct our hypothesis by setting $l_h = max(x_1^t, x_2^t)$ because the hypothesis will always make mistakes on all values of $l_h < max(x_1^t, x_2^t)$

ii. Negative Example:
When there is a mistake on positive example, we have label $y^t = -1$ and equation $2l - |x_1^t| - |x_2^t| \geq 0$. For negatiive example i.e $y^t = -1$ we must have either x_1^t, x_2^t or both greater than value of l_c for the correct function. l_h is higher than all of these values, therefore we set $l_h = max(x_1^t, x_2^t) - 1$. This will ensure our hypothesis matches the label for current example.

(d) Let the $l = l_h$ for current hypothesis. {..} represents comments in the pseudocode.

Algorithm 1 Mistake driven algorithm to learn correct function $f \in C$

Start with $l_h = 40$ {We set l_h to half of the range of l}
for each example **do**
  if $y^t(2l - |x_1^t| - |x_2^t|) < 0$ **then**
    if $y^t == +1$ **then**
      $l_h = max(x_1^t, x_2^t)$ {If mistake is made on positive example}
    else
      $l_h = max(x_1^t, x_2^t) - 1$ {If mistake is made on negative example}
    end if
  end if
end for

On any given dataset, the algorithm will make maximum of m mistakes where

$$m = |40 - l_c|$$

where l_c is value of length l for correct function $f \in C$.

2. In the halving algorithm we predict output which agrees with majority of functions in the current concept class C. If a mistake is made, we remove the functions which predicted the wrong output and repeat the process on next example. This leads to cutting down of experts by atleast half each time a mistake is made. This process stops when we reach down to 1 expert because he has predicted correct output for all the examples. Now if we have M experts in the initial concept class C we will stop the algorithm when we have cut down the pool of experts to these final M functions.
Let $|C| = N$ be the initial size of concept class. When 0 mistakes are made we have size of initial concept class $|C_0| = |C|$. When first mistake is made,

$$|C_1| \leq \frac{1}{2}|C_0|$$

When 2 mistakes are made,

$$|C_2| \leq \frac{1}{2^2}|C_0|$$

. Let n be the number of mistakes for which the size of concept class is cut down to M. Therefore,

$$M = |C_n| \leq \frac{1}{2^n}|C_0|$$

Now in worst case we commit $M = \frac{1}{2^n}|C_0|$ mistakes. But $|C_0| = |C| = N$. Therefore,

$$M = \frac{1}{2^n}N$$

$$\therefore 2^n = \frac{N}{M}$$

$$\therefore n = \log_2\left(\frac{N}{M}\right)$$

$\therefore$ Mistake bound of halving algorithm in case of M experts is $O(\log \frac{N}{M})$

3 The Perceptron Algorithm and its Variants

1.

2. The number of instances of each label in training set:

$$1 = 4606$$

$$-1 = 3685$$

The number of instances of each label in dev set:

$$1 = 759$$

$$-1 = 623$$

The number of instances of each label in test set:

$$1 = 792$$

$$-1 = 590$$

Let us consider a classifier which predicts the most frequent label. So this classifer will train will always predict the majority label 1.

Accuracy on dev set:
$A_{dev} = 759/(759 + 623) * 100 = 54.92\%$

Accuracy on test set:
$A_{test} = 792/(792 + 590) * 100 = 57.30\%$

3.