CSE 4502/5717 Big Data Analytics. Fall 2022

Exam III Solutions

1. Note that for a pair of items to be frequent, it has to occur in at least one transaction. Thus the only two itemsets we have to generate as candidates are the pairs of items we can generate from each transaction. As a result, the number of candidates is $O(n)$. For each such candidate we have to compute the support.

   Let the candidates be $(i_1^1, i_2^1), (i_1^2, i_2^2), \ldots, (i_1^q, i_2^q)$, where $q = O(n)$. We keep any such 2-itemset in sorted order (i.e., $i_1^k < i_2^k$, for $1 \leq k \leq q$). We can think of each such candidate as an integer in the range $[1, n^{2c}]$. We sort these 2-itemsets in lexicographic order using the integer sorting algorithm. This will take $O(n)$ time. We scan through the sorted sequence, identify all the frequent 2-itemsets, and output them.

2. Note that for an itemset to be frequent, it has to occur in at least one transaction. Thus the only itemsets we have to generate as candidates are itemsets that we can generate from each transaction. From each transaction, we can generate at most $2^c - 1$ itemsets. As a result, the number of candidates is $O(n)$. For each such candidate we have to compute the support.

   Support for the candidates can be computed as described in class. In particular, we use a hash tree to store all the candidates. The expected size of each leaf in the hash tree is $O(1)$. We output all the itemsets that have a support of $\geq minSupport$. It is clear that the above algorithm has an expected run time of $O(n)$.

3. Use divide and conquer technique. Compute $(x+a_1)(x+a_2)\ldots(x+a_{n/2})$ and $(x+a_{n/2+1})(x+a_{n/2+2})\ldots(x+a_n)$ recursively and multiply the resulting polynomials. If $T(n)$ is the complexity to compute $f(x)$ then $T(n) = 2T(n/2)+O(n\log n)$, which implies $T(n) = O(n\log^2 n)$.

4. Convert the polynomial $f(x)$, into the following form (into n/m groups):
   $$\begin{aligned} f(x) = {} & x^{n-m}(a_n x^m + a_{n-1}x^{m-1} + \ldots + a_{n-m+1}x) + \\ & x^{n-2m}(a_{n-m}x^m + a_{n-m-1}x^{m-1} + \ldots + a_{n-2m+1}x) + \\ & \vdots \\ & (a_m x^m + a_{m-1}x^{m-1} + \ldots + a_1 x + a_0) \\ = {} & x^{n-m}f_1(x) + x^{n-2m}f_2(x) + \ldots + f_{n/m}(x) \end{aligned}$$
   where $f_1(x)$, $f_2(x)$, …, $f_{n/m}(x)$ are polynomials of degree m.
   $f(x)g(x) = x^{n-m}f_1(x)g(x) + x^{n-2m}f_2(x)g(x) + \ldots + f_{n/m}(x)g(x)$
   $f(x)g(x)$ requires multiplying (n/m) pairs of polynomials of degree m each and combining the n/m resulting polynomials into one polynomial of degree $n+m$. The second step requires a scan of all the n/m polynomials of degree $2m$.
   Total time taken is $O(\frac{n}{m}(m\log m)) + O(\frac{n}{m}(2m)) = O(n\log m)$.

5. The loss function is $L(w_1, w_2) = (w_2 - 3)^2 + (w_1 - 4)^2 + (w_1 + w_2-)^2 + (2w_1 + w_2 - 10)^2$ $= 6w_1^2 + 3w_2^2 + 6w_1w_2 - 60w_1 - 38w_2 + 161$. We want to have: $\frac{\partial L}{\partial w_1} = 0$ and $\frac{\partial L}{\partial w_2} = 0$.

$\frac{\partial L}{\partial w_1} = 0$ implies that $12w_1 + 6w_2 = 60$ and $\frac{\partial L}{\partial w_2} = 0$ implies that $6w_1 + 6w_2 = 38$. Solving these two equations, we get: $w_1 = \frac{11}{3}$ and $w_2 = \frac{8}{3}$.

6. Here is a multilevel perceptron for realizing the Boolean function $F(x_1, x_2, x_3, x_4) = x_2x_3 + \bar{x_1}\bar{x_4} + x_2\bar{x_3}\bar{x_4}$:

Figure 1: A neural network for $F(x_1, x_2, x_3, x_4) = x_2x_3 + \bar{x_1}\bar{x_4} + x_2\bar{x_3}\bar{x_4}$.