Algorithms for Data Science: Web Advertising Algorithms

Pablo Mollá Chárlez

October 23, 2024

Contents

1 Exercise 1: Frequent Items, Association Rules, A-Priori (8 points)

Assume we have a market basket model, having 100 items, numbered from 1 to 100, and 270 baskets, numbered from 1 to 270. Items i is in basket b if b divides $3i$ with no remainder. For instance, item 1 is present in all baskets which are multiples of 3.

Questions:

1. How many items are in a basket b? How many baskets have items? Give a formula or explain it in English.
2. Which are the items having support thresholds 1 and 5? You do not need to write all of them; you can just give a description and justification for each case.
3. What are the pairs of items having support threshold 5? You do not need to write all of them; you can just give a description and justification for each case. Explain how you constructed the pairs of items.
4. Give the confidence of the association rules $\{2, 4, 6\} \longrightarrow 1$ and $\{2, 3\} \longrightarrow 5$.
5. Explain, step by step, the functioning of the A-Priori algorithm for support threshold. At each step, explain the data structures you used and how they were constructed.

1.1 Solution

- **Question 1**

  Any basket with $j \in \{1, \ldots, 270\}$ can be defined as $B_j = \{i \mid 3i \; mod \; j \;\cong\; 0\}$ where $i \in \{1, \ldots, 100\}$. Therefore, the basket B_j for a given j contains as many elements as we can express as $i = \frac{j \cdot k}{3}$ where $k \in \mathbb{Z}, j \in \{1, \ldots, 270\}$ and $i \in \{1, \ldots, 100\}$. For a basket B_j with $j \in \{1, \ldots, 270\}$ to have items, j must be a multiple of 3, i.e. $j = 3 \cdot k'$ where $k' \in \mathbb{Z}$ with $1 \leq j = 3 \cdot k' \leq 270$. The highest basket number is 270, meaning that $270 = 3 \cdot k' \longleftrightarrow k' = 90$. Therefore, there are 90 baskets with items (non-empty baskets).

- **Question 2**

  An item $i \in \{1, \ldots, 100\}$ will be in a basket j if $j \mid 3i$ (j divides $3i$). We previously established that j must be a multiple of 3 for this to hold. Therefore $j = 3k$ for some $k \in \mathbb{Z}$, and the condition then becomes $(3k) \mid 3i \iff k \mid i$. This means that for a basket j to contain an item i, the corresponding k ($j = 3k$) must be a divisor of i. Then, the support of item i is the number of divisors of i.

  Therefore, an item i has support threshold 1 if it appears in exactly 1 basket, i.e, i has just one divisor. Similarly, an item i has support threshold 5 if it appears exactly in 5 baskets, i.e, i has exactly 5 divisors.

- **Question 3**

  The pairs of items having threshold 5 are the ones with 5 divisors. For instance, we could use a property of prime numbers that states that for any prime number p, the number of divisors of p^n is $(n+1)$. Therefore, by selecting for example $2^4 = 16$ with 2 being the prime number, we have 5 divisors $\{1, 2, 4, 8, 16\}$, meaning that the item 16 will be present in exactly 5 baskets, presumably $B_3, B_6, B_{12}, B_{24}, B_{48}$ (you just need to consider $3 \cdot i, i \in \{1, 2, 4, 8, 16\}$.

- **Question 4**

  The confidence of a given association rule $A \longrightarrow B$ is defined as: $conf(A \longrightarrow B) = \frac{supp(A,B)}{supp(A)}$. In our case, we need to determine the supports of $\{2, 4, 6\}, \{1, 2, 4, 6\}, \{2, 3\}$ and $\{2, 3, 5\}$. First, we analyze the support for the individual itemsets and then we deduce the support of the larger itemsets.

  For instance, 1 belongs to a basket $B_j \iff 3 \ mod \ j \cong 0$, meaning that j divides 3, and therefore either $j = 1$ or $j = 3$. Following this reasoning, we obtain the following values for j:

  - For item 1 $\implies 3(1) \ mod \ j \cong 0 \longleftrightarrow j \in \{1, 3\}$
  - For item 2 $\implies 3(2) \ mod \ j = 6 \ mod \ j \cong 0 \longleftrightarrow j \in \{1, 2, 3, 6\}$
  - For item 3 $\implies 3(3) \ mod \ j = 9 \ mod \ j \cong 0 \longleftrightarrow j \in \{1, 3, 9\}$
  - For item 4 $\implies 3(4) \ mod \ j = 12 \ mod \ j \cong 0 \longleftrightarrow j \in \{1, 2, 3, 4, 6, 12\}$
  - For item 5 $\implies 3(5) \ mod \ j = 15 \ mod \ j \cong 0 \longleftrightarrow j \in \{1, 3, 5, 15\}$
  - For item 6 $\implies 3(6) \ mod \ j = 18 \ mod \ j \cong 0 \longleftrightarrow j \in \{1, 2, 3, 6, 9, 18\}$

  Then, for the supports of $\{2, 4, 6\}$ and $\{1, 2, 4, 6\}$ we deduce that the common j values that would include the 3 and 4 items are respectively $j = 1, 3$ (2 baskets, B_1 and B_3) and $j = 1, 2, 3, 6$ (4 possible baskets, B_1, B_2, B_4, B_6). Similarly, for itemsets $\{2, 3\}$ and $\{2, 3, 5\}$ we respectively obtain in both cases $\{1, 3\}$. We can conclude that:

  * $conf(\{2, 4, 6\} \longrightarrow \{1\}) = \frac{supp(\{1,2,4,6\})}{supposed(\{2,4,6\})} = \frac{\frac{2}{270}}{\frac{4}{270}} = \frac{1}{2}$.
  * $conf(\{2, 3\} \longrightarrow \{5\}) = \frac{supp(\{2,3,5\})}{supposed(\{2,3\})} = \frac{\frac{2}{270}}{\frac{2}{270}} = 1$.

- **Question 5**

  The Apriori Algorithm identifies frequent itemsets by making use of the monotonicity principle, which states that if an itemset is frequent then all its subsets must also be frequent. It operates in 2 main passes:

  - **Pass 1**: A hash tables stores the counts of individual itemsets. The algorithm scans the whole database and counts the occurences of each item in all baskets, then retains items whose support meets the given threshold s and outputs the frequent $1 - itemsets$.
  - **Pass 2**: To find $2 - itemsets$ and larger $k - itemsets$, we can use a triangular array to store the counts of $2 - itemsets$ (2D pairs (i, j) are indexed into 1D array with index $k = (i-1)(n - \frac{n}{2} + (j - i)$ for i, j) and then prune the infrequent pairs according to the threshold. We then repeat the process by extending the frequent $2 - itemsets$ to generate $3 - itemsets$ and then prune once again. The monotonicity principle is key, as we only extend frequent itemsets.

2 Exercise 2: Counting Ones in a Window (7 points)

Consider the following stream of bits, where the rightmost element is the most recent one:

$$\ldots\ 1\ 1\ 1\ 1\ 0\ 0\ 1\ 0\ 1\ 1\ 1\ 0\ 0\ 1$$

We aim to estimate the number of 1s in the last k bits using the DCIM algorithm.

Questions:

1. Assume DCIM has created the following buckets, where the rectangles represent the buckets of increasing number of 1s inside:

   $$\ldots\ \boxed{1\ 1\ 1\ 1}\ 0\ 0\ \boxed{1\ 0\ 1}\ \boxed{1\ 1}\ 0\ 0\ \boxed{1}$$

   Explain how DCIM estimates the number of 1s from these buckets, for $k = 5$ and $k = 11$. If it is the cases, explain why the estimation of the number of 1s is wrong.

2. Give one other way to divide the window into buckets, while respecting the DCIM restrictions on buckets.

3. Assume the following bits arrive in the stream in order: 0, 1, 0, 0, 1, 1. Explain how the buckets are updated from the setting in Question 2.1 and what is the final result.

4. Take the following stream of integers between 0 and 3, where the rigthmost element is the most recent one:

   $$\ldots\ 2\ 3\ 2\ 1\ 1\ 1$$

   Explain and exemplify how you can use 2 streams and DCIM to estimate the sum of the last 4 elements of the stream.

2.1 Solution

- **Question 1**

  The DGIM algorithm is designed to find the number of 1's in a stream of data or dataset. This algorithm uses $O(log^2(N))$ bits to represent a window of N bits, and it allows to estimate the number of 1's with an error of no more than 50%. In our situation, we have a stream of bits with the following buckets where the rightmost element is the most recent one:

  $$\boxed{1\ 1\ 1\ 1}\ 0\ 0\ \boxed{1\ 0\ 1}\ \boxed{1\ 1}\ 0\ 0\ \boxed{1}$$

  For $\boxed{\text{k=5}}$, our goal is to estimate the number of 1's in the last 5 bits of the stream, i.e.: $\boxed{1\ 1}\ 0\ 0\ \boxed{1}$. The last bucket $\boxed{1}$ includes 1 one, corresponding to the 1 at the last position in the stream. This bucket fits completely within the last 5 bits, therefore we include its count for the estimation. For the next bucket, $\boxed{1\ 1}$, which contains 2 ones, both of these 1's also fit completely within the last 5 bits, therefore we count 2 for the estimation. The final estimation is then $1 + 2 = 3$ which matches the actual count of 1's.

  For $\boxed{\text{k=11}}$, our goal is to estimate the number of 1's in the last 11 bits of the stream, we highlight with colours the corresponding 1's (zeros aren't) within the last 11 bits:

  $$\boxed{1\ 1\ 1\ 1}\ 0\ 0\ \boxed{1\ 0\ 1}\ \boxed{1\ 1}\ 0\ 0\ \boxed{1}$$

  The last bucket $\boxed{1}$ includes 1 one, corresponding to the 1 at the last position in the stream, and as previously, fits completely within the last 11 bits. Following the same reasoning, the buckets $\boxed{1\ 1}$ and $\boxed{1\ 0\ 1}$ also fit within the 11-long bits of array. However, when it comes to the last bucket (first starting from left) $\boxed{1\ 1\ 1\ 1}$, only a part of it (last 1 of the bucket is included in the 11 bits), therefore according to the DGIM algorithm procedure, instead of counting the total number of 1's in such bucket, we count half of it, meaning we count just 2 ones instead of 4. Thus, the total estimation of 1's, in the last 11 bits of the stream results in a total of: $2 + 2 + 2 + 1 = 7$, which is a very close estimation to the actual value of 6.

- **Question 2**

  There is no other way to divide the window into buckets that creates buckets with size in power of 2 or in increasing size from right to left. The requirements for the buckets are the following:

  Requirements

  1. Right side of the bucket should always start with 1.
  2. Every bucket should have at least one 1.
  3. All buckets should have a number of 1's in power of 2.
  4. Buckets can't decrease in size as we move to the left side.
  5. The buckets should have at most 2 buckets of size 1.

- **Question 3**

  We assume that we have a new stream of bits arriving to the proposed new division of buckets and we study how the buckets are modified by the algorithm:

  $$\boxed{1\,1\,1\,1}\,0\,0\,\boxed{1\,0\,1}\,\boxed{1\,1}\,0\,0\,\boxed{1} \longleftarrow 0\,1\,0\,0\,1\,1$$

  For the first bit 0, no modifications need to be implemented, and for the second arriving bit 1, we create a new bucket to contain it, therefore updating the buckets structure as follows:

  $$\boxed{1\,1\,1\,1}\,0\,0\,\boxed{1\,0\,1}\,\boxed{1\,1}\,0\,0\,\boxed{1}\,0\,\boxed{1} \longleftarrow \; 0\,0\,1\,1$$

  In this situation, the algorithm tells us that we can have at most 2 buckets of size 1, thus no problem. Then, the third and fourth bit come, but as they are zeros, no modifications are considered. However, for the last 2 bits we apply modifications, the fifth bit forces us to create a new bucket $\boxed{1}$ which is added to the complete structure:

  $$\boxed{1\,1\,1\,1}\,0\,0\,\boxed{1\,0\,1}\,\boxed{1\,1}\,0\,0\,\boxed{1}\,\boxed{1}\,0\,0\,\boxed{1} \longleftarrow 1$$

  Now, we have 3 buckets of size 1, forcing us to merge the 2 previous ones ($\boxed{1}\,\boxed{1}$) as a single bucket of size 2 ($\boxed{1\,1}$). And finally, we create the last bucket for the bit 1:

  $$\boxed{1\,1\,1\,1}\,0\,0\,\boxed{1\,0\,1}\,\boxed{1\,1}\,0\,0\,\boxed{1\,1}\,0\,0\,\boxed{1}\,\boxed{1}$$

  Now, we have 3 buckets of size 2, forcing us to merge the 2 previous ones ($\boxed{1\,0\,1}\,\boxed{1\,1}$) as a single bucket of size 4 ($\boxed{1\,1}$), i.e. : $\boxed{1\,1\,1\,1} = \boxed{1\,0\,1} + \boxed{1\,1}$

  $$\boxed{1\,1\,1\,1}\,0\,0\,\boxed{1\,1\,1\,1}\,0\,0\,\boxed{1\,1}\,0\,0\,\boxed{1}\,\boxed{1}$$

- **Question 4**

  We will consider 2 streams of data to answer the question. For the first stream, we use the DGIM algorithm to estimate the number of 2's and 3's which will be considered as 1 and the actual ones as 0:

  $$\ldots\; 1\,1\,1\,0\,0\,0$$

  Similarly, we apply the DGIM algorithm for a second stream to estimate the number of 1's which will remain as 1's but the 2's and 3's become 0:

  $$\ldots\; 0\,0\,0\,1\,1\,1$$

  After each estimation, we would have on one hand, an estimation of the number of 2's and 3's, and on the other hand, an estimation of the number of 1's in the stream. Now, we just need to multiply the first stream estimation by 2 or 3, and add it to the second estimation of 1's to obtain the total estimation sum.

3 Exercise 3: Counting Distinct Items (5 points)

Consider the following stream:

3 1 1 1 2 3 3 5 5 1

We want to estimate the number of distinct items using several hash functions of the form $ax + b \ mod \ 32$, i.e., hash the numbers on 5 bits. We want to use the estimate of Flajolet-Martin, which uses the number of tail $0s$ in the hash representation to estimate the number of distinct items in the stream. For each of the following hash functions (h_1, h_2 and h_3) and the above stream explain the functioning of the Flajolet-Martin algorithm and the estimation of the number of distinct items. Compare it with the real number of distinct items in the above. How would you improve the estimation? Exemplify. The hash functions are:

$$h1(x) = 3x + 1 \ mod \ 32 \qquad h2(x) = x + 2 \ mod \ 32 \qquad h3(x) = 5x + 7 \ mod \ 32$$

3.1 Solution

Let's first explain the Flajolet-Martin (FM) algorithm and then apply it to the defined scenario. The FM algorithm allows to determine an estimation of the number of distinct elements within a stream of data by using hash functions. The steps of the FM algorithm involve:

1. **Hash each element of the stream using each of the hash functions.**
2. **Convert each result of the hash functions into a 5-bit binary numbers.**
3. **Count the number of trailing zeros for each element in each hash function.**
4. **Extract the maximum number of trailing zeros from each hash function to estimate the number of distinct elements, either by considering the average, median or grouped medians of the results.**

To improve the algorithm's results, we would need to increase the number of hash functions used, we could as well use multiple independent streams of data or directly consider a more refined version of the FM algorithm such as the HyperLogLog Algorithm. Now, let's solve the exercise by following the steps previously described:

- $h1(1) = 3(1) + 1 \ mod \ 32 = 4 = (00100)_5 \implies$ 2 trailing zeros.
- $h1(2) = 3(2) + 1 \ mod \ 32 = 7 = (00111)_5 \implies$ 0 trailing zeros.
- $h1(3) = 3(3) + 1 \ mod \ 32 = 10 = (01010)_5 \implies$ 1 trailing zeros.
- $h1(5) = 3(5) + 1 \ mod \ 32 = 16 = (10000)_5 \implies$ 4 trailing zeros.
- $h2(1) = (1) + 2 \ mod \ 32 = 3 = (00011)_5 \implies$ 0 trailing zeros.
- $h2(2) = (2) + 2 \ mod \ 32 = 4 = (00100)_5 \implies$ 2 trailing zeros.
- $h2(3) = (3) + 2 \ mod \ 32 = 5 = (00101)_5 \implies$ 0 trailing zeros.
- $h2(5) = (5) + 2 \ mod \ 32 = 7 = (00111)_5 \implies$ 0 trailing zeros.
- $h3(1) = 5(1) + 7 \ mod \ 32 = 12 = (01100)_5 \implies$ 2 trailing zeros.
- $h3(2) = 5(2) + 7 \ mod \ 32 = 17 = (10001)_5 \implies$ 0 trailing zeros.
- $h3(3) = 5(3) + 7 \ mod \ 32 = 22 = (10110)_5 \implies$ 1 trailing zeros.
- $h3(5) = 5(5) + 7 \ mod \ 32 = 0 = (00000)_5 \implies$ 5 trailing zeros.

We conclude that the estimation of distinct elements from h_1 is $2^4 = 16$, from h_2 is $2^2 = 4$ and from h_3 is $2^5 = 32$. If we consider the average, we would obtain an estimation of approximately 17 different items and with the median we would have 16 different items.