CS60021: Scalable Data Mining 2019
Sample Questions: Reservoir Sampling and Bloom Filters

1. Suggest an algorithm for uniform sampling from distributed streams.

2. What does Bloom filter tell us about an item?

   A. An item is definitely in the set or may be not in the set.

   B. An item may or may not be in the set.

   C. An item is definitely not in the set or may be in the set.

   D. An item is definitely in the set.

3. Consider a following Bloom Filter implementation: Let $S = \{s_1, s_2, \ldots, s_m\}$ be the universe. There is an array A of n bits, $A[0]$ to $A[n-1]$. And there are k independent hash functions $h_1, h_2, \ldots, h_k$, each with a range $\{0, 1, 2, \ldots, n-1\}$. We assume that each of the hash functions map each element in the universe U to a random number uniformly in the range $\{0, 1, 2, \cdots, n-1\}$. For each $s \in S$ we set to one all the bits $A[h_i(s)]$, for $i = 1, \ldots, k$. Which of the following statements are *True* when you increase k, the number of hash functions.

   A. there exists a value k_0 such that until k reaches k_0, increasing k actually decreases the probability of false positive.

   B. there exists a value k_0 such that, if $k > k_0$, increasing k causes probability of false positive to increase.

   C. with higher k, more checks are required, and thus the probability of false negative increases.

   D. with higher k, more and more ones are set in the array and thus the probability of false negative decreases.

4. Consider the Bloom Filter implementation from Q2. After all the elements from S are hashed into the Bloom filter, What is the probability that a specific bit remains 0?

   A. $\left(1 - \frac{1}{n}\right)$

   B. $\left(1 - \frac{1}{m}\right)^{kn}$

   C. $\left(1 - \frac{1}{n}\right)^{km}$

   D. $\frac{1}{n}$

5. Lets say there are n students $\{s_1, s_2, \ldots, s_n\}$ in the course and they want to use a central server. We create a hash function, that operates on user-id i.e. student s_i is hashed to $h(s_i)$. We plan to give access to the server in the some ordering (say sorted) of the hash values returned by $h(s_i)$. Lets say that the hash value is of b bits and can be considered to be choosing values uniform at random. Given two fixed users, what is the probability that they get the same hash value?

A. $\left(\frac{1}{n}\right)$

B. $\left(\frac{1}{n^b}\right)$

C. $\left(1-\frac{1}{n}\right)$

D. $\left(\frac{1}{2^b}\right)$

6. Use the information from the above question (i.e. Q4). What is the probability that at least one pair of students share a same hash value?

A. $1-\frac{n(n-1)}{2}\left(1-\frac{1}{2^b}\right)$

B. $1-\left(1-\frac{1}{2^b}\right)^{\frac{n(n-1)}{2}}$

C. $\left(1-\frac{1}{2^b}\right)^{\frac{n(n-1)}{2}}$

D. $1-\left(1-\frac{1}{2^b}\right)$

7. Use the information from the above two questions (i.e. Q4 and Q5). We know that if we use more bits (i.e. higher value of b), then with high probability the students would get distinct hash values. We cannot use infinitely many bits, but we are fine if the probability that a pair of students get same hash value is very low i.e. $1/n$. So, how many bits of hash value should we use so that the the probability that any pair of students get same hash value is at most $1/n$?

A. No way, you have to take infinitely many bits

B. $b \geq n$

C. $b \geq 3\log_2\ n$ is enough.

D. $b \geq n^2$

8. Deletion of elements from Bloom filter is not allowed. Why?

A. It will increase the false positive rate.

B. Bloom filters are immutable.

C. It leads to deleting other elements hashed to same indices, leading to false negatives.

D. It leads to shrinking of filter size.

9. Suppose we are trying to create "ideal filter". How many bits (b) are necessary to represent all the sets of N elements from the universe of size U allowing false positives for at most a fraction λ of the universe and no false negatives? (Hint: Each b-bit string for the ideal filter must accept all N elements and at most $\lambda(U-N)$ non-elements. There should be at least one b bit for every set of size N. How many such b-bit strings need be there?)

(a) $b = N\ \log_2(\lambda)$

(b) $b = N\ \lambda$

(c) $b = N\ \log_2\ \left(\frac{1}{\lambda}\right)$

10. You have a huge dataset of fingerprints for which you want to create a Bloom filter. The dataset is distributed on 10 different machines. Let us denote the dataset at machine i to be M_i. Your goal is to maintain a array of m bits and use k hash functions for each of the fingerprints. The set of k hash functions are the same in all the machines.

    Consider the following two strategies of creating the Bloom filters:
    Strategy-1:

    - For each machine i create a local Bloom filter B_i
    - return B_U = bitwise OR of B_i's

    Strategy-2:

    - Take the union of fingerprints $M_U = \cup_{i=1}^{10} M_i$ by gathering the data on one machine
    - return Bloom filter $\mathcal{B}_{\mathcal{U}}$ created using data M.

    Which of the following statements are *True*:

    A. B_U will have a higher false positive rate as compared to $\mathcal{B}_{\mathcal{U}}$ as the number of 1's in B_U is at least the number of 1's in B_i $(i = 1, 2, \dots, 10)$

    B. B_U will be same as $\mathcal{B}_{\mathcal{U}}$

    C. B_U will have a lesser false positive rate as compared to $\mathcal{B}_{\mathcal{U}}$ as the number of 1's in B_U is at least the number of 1's in B_i $(i = 1, 2, \dots, 10)$

    D. We cannot say anything about the false positive rates of both the Bloom filters

11. Consider the information from the previous question (i.e. Q9). And here are couple of more strategies for creating the Bloom filter.

    Strategy-3:

    - For each machine i create a local Bloom filter B_i
    - return B_I = bitwise AND of B_i's

    Strategy-4:

    - Take the intersection of fingerprints $M_I = \cap_{i=1}^{10} M_i$ by gathering the data on one machine
    - return Bloom filter $\mathcal{B}_{\mathcal{I}}$ created using data M_I.

    A. B_I will have a higher false positive rate as compared to $\mathcal{B}_{\mathcal{I}}$ as the number of 0's in B_I is at least the number of 0's in B_i $(i = 1, 2, \dots, 10)$

    B. The false positive rate in B_I is at most the false positive rate in each of B_i $(i = 1, 2, \dots, 10)$, but can be larger than the false positive rate of $\mathcal{B}_{\mathcal{I}}$

    C. B_I will have a lesser false positive rate as compared to $\mathcal{B}_{\mathcal{I}}$ as the number of 0's in B_I is at least the number of 0's in B_i $(i = 1, 2, \dots, 10)$

    D. We cannot say anything about the false positive rates of both the false positive rates.

    E. B_I will be same as $\mathcal{B}_{\mathcal{I}}$