Data Mining — August 23, 2018
IIIT Delhi — CSE 560, Monsoon 2018
Sumit Bhatia — Assignment

Assignment 1
Due Wednesday, 11:59PM, September 5, 2018

Policy

- This assignment is to be done individually. You will be required to demo your code to the TAs.
- Any help taken from any outside source *must* be reported in your report.
- Institute's policy on plagiarism will apply.
- Last date for doubt clearance: Friday, August 30, 4 PM.
- Late submissions penalized by 10% every three hours.

Sampling a continuous data stream: Welcome back to the course after a gap of about two weeks. In the last lecture, we studied about sampling and how it can be used to obtain a much smaller, representative sample of the data to be analyzed. Through this exercise, we will learn how to sample data from a large, continuous stream of data.

Imagine you are working in the data science team of an advertising agency. Asian Games are going on and being the data nerd you suggest that it is a great opportunity for the company to understand the characteristics of consumers interested in sports. For example, what is the average age of users interested in specific sports and what are their other interests. Having such insights, you can engage with users in real time and show them relevant ads. While you blame your instructor for not covering the association rule mining part yet, you feel confident about the first part.

Armed with your recently acquired knowledge of sampling, you suggested that we should sample the users tweeting about Asian Games and study their characteristics. It will clearly require less resources to study properties of a representative sample, right? This is important for a real time system. Also, since potentially there could be millions of such tweets, you only want to store a small number of representative tweets. Impressed, your boss asked you to do that.

Feeling excited, you decide to create a uniform sample of all the tweets containing the hashtag *#asiangames2018*. You quickly wrote a piece of software that "listens" to twitter and gets all the tweets containing the hashtag. For sampling, you just need to retain only a subset of tweets your software is gathering. Since you need to create an unbiased sample, *each tweet should have an equal chance of being selected in the sample.* So to create a sample of, say 20,000 tweets, you just need to select 20,000 tweets at random from all the tweets you get and you will be done. Happily, you started writing the code and then...

You realized that in order to create a uniform sample of 20,000 tweets, you first need to know how many tweets are actually there!

Let the fun begin!

1. Assuming you have magical powers and you know how many tweets will be made containing the hashtag *#asiangames2018*. Let N be the number of such tweets. Describe how you can obtain a uniform sample of size $s (s \ll N)$. **(10 points)**

2. Coming back to the real world where you have no magical powers, N is not known to you. In fact, you get a continuous stream of tweets containing the desired hashtag. Let us see how can we solve this problem. Let us assume that we want to create a sample of size s. Write a code (java/python) that does the following:

   (a) Write a method *getNextStream(int n)* that returns an array of size n. The n elements of the array are a random permutation of the set $\{A_1, A_2, \ldots, A_n\}$, where n is the size of the stream. This method can be used to simulate a continuous data stream of varying sizes. **(20 points)**

   (b) Once you have a stream of data generated above, we now create a *sample* array of size s to hold items of our sample. Write a method *updateSample(String streamItem, int itemNumber)* that accepts a streamItem (one of the item from the stream generated above) and the position/sequence of the item in the stream. For example, if the 5^{th} item in the stream is A_9, you will call updateStream(A_9,5). This method then performs following two steps: **(50 points)**

      i. If $itemNumber <= s$; add *streamItem* into your sample

      ii. If $itemNumber > s$; select *streamItem* to be included into your sample (size s) with a probability $(s/itemNumber)$.

      iii. If *streamItem* is selected to be included in the sample, remove one of the s existing items in the sample at random (i.e., all items already existing in the sample can be evicted with equal probability).

   (c) Using the above two methods, write a method/function that *(i)* creates a stream of size n by calling *getNextStream(n)*; and *(ii)* passes each stream item through *updateSample* to create a sample of size s. **(30 points)**

   (d) Use the above generated methods to create a sample of size $s = 5$ from a data stream of size $n = 20$. Repeat this sampling process for $N = \{100, 500, 1000, 10000\}$ times and count the number of times each item from the stream is included in the sample and report your observations in following format. **(40 points)**

	N=100	N=500	N=1000	N=10000
A_1	$count(A_1)$	$count(A_1)$	$count(A_1)$	$count(A_1)$
..				
A_{20}	$count(A_{20})$	$count(A_{20})$	$count(A_{20})$	$count(A_{20})$

   (e) Describe your observations about the counts in above table. Based on the results, is the sampling process you just implemented biased or unbiased? **(10 points)**

   (f) **Extra Credit:** Prove that after n stream points have arrived, the probability of any stream point being included in the sample of size s as created by the process described in 2(b) is same; and is equal to s/n. **(40 points)**