Reservoir Sampling

By Shuangjiang Li (June 06, 2014)

1. 介绍

> Reservoir Sampling 中文翻译是水塘抽样/蓄水池抽样， 指的是一组随机抽样算法， 而不是某一个具体的算法。 这类算法主要用于解决这样一个问题： 当样本总体很大或者是在数据流上进行采样的时候， 我们往往无法预知总体的样本实例个数 n 。 那么 Reservoir Sampling 就是这样一组算法， 即使不知道 n， 也能保证每个样本实例被采样到的概率依然相等。 下面是一个其中一个非常典型的在线(online)线性算法， 可以用于数据无法装入内存或者在流数据上使用。

2. 实现

从 n 个数据里随机取 k 个数：

1. 用前 k 个数据组成一个reservoir;
2. 从 $k+1$ 开始，对于第 i 个元素，随机产生一个 $[1,i]$ 之间的数 j，如果 $j \leq k$，则reservoir里第 j 个元素就被 i 替换掉。直到 n 个数据都被试过。

3. 证明

- 上面这个算法的步骤是这样:

    1. 从总体 n 中抽取前面的 k 个实例放入预制的数组中， 这个数组就是我们最后要返回的抽样结果;

    2. 对于后面的所有样本实例， 从 $i = k+1$ 个开始， 我们对每一个生成一个 $[0,i)$ 的随机数 rnd， 若 $rnd < k$， 那我们就用当前的val替换掉 result[i]。

- 这样做为什么能保证每个实例被抽到的概率当等而且概率为 $\frac{k}{n}$ 呢?

可以这样分析: 我们知道对于第 i 个实例， 当算法遇到他的时候， 他被选中进入 result 的概率是 $\frac{k}{i}$， 那么他依然出现在最后的 result 的情况是， i 后面所有的实例都没有取代掉他， i 后面任何第 $t > i$ 个实例取代掉他的概率是 $(\frac{\frac{k}{t}}{k})$。 **可以这样理解， $\mathbf{t}$ 实例被选中的概率 $(\frac{\mathbf{k}}{\mathbf{t}})$， 选中而且取代原来 $\mathbf{i}$ 所在的位置的概率即是 $(\frac{\frac{\mathbf{k}}{\mathbf{t}}}{\mathbf{k}})$**。所以后面任意一个实例不取代 i 的概率就是 $1-\frac{1}{t}$， 那么要所有的的情况都发生， 最后 i 才能留在result中， 这样就是一个连乘的结果：

$$(\frac{k}{i}) * (1-\frac{1}{i+1}) * (1-\frac{1}{i+2}) \ldots * (1-\frac{1}{n})) = \frac{k}{n}$$

4. Java 代码

```
 1. public static int[] reservoirSampling(int[] data, int k) {
 2.         if (data == null) {
 3.             return new int[0];
 4.         }
 5.
 6.         if (data.length < k) {
 7.             return new int[0];
 8.         }
 9.
10.         int[] sample = new int[k];
11.         int n = data.length;
12.
13.         for (int i = 0; i < n; i++) {
14.             if (i < k) {
15.                 sample[i] = data[i];
16.             } else {
17.                 int j = new Random().nextInt(i);
18.                 if (j < k) {
19.                     sample[j] = data[i];
20.                 }
21.             }
22.         }
23.         return sample;
24.
25.     }
26.
27.     public static void main(String[] args) {
28.         int k = 100;
29.         int n = 1000;
30.
31.         int[] data = new int[n];
32.         for (int i = 0; i < n; i++) {
33.             data[i] = i;
34.         }
35.
36.         int[] sample = reservoirSampling(data, k);
37.         System.out.println(Arrays.toString(sample));
38.     }
39.
```