Asmt 5: Frequent Items

Turn in through Canvas by 2:45pm:
Wednesday, Feb 26
100 points

Overview

In this assignment you will explore finding frequent items in data sets, with emphasis on streaming techniques designed to work at enormous scale. For simplicity you will work on more manageably sized data sets, and simulate the stream by just processing with a for loop.

You will use two data sets for this assignment:

- `http://www.cs.utah.edu/~jeffp/teaching/cs5140/A5/S1.txt`
- `http://www.cs.utah.edu/~jeffp/teaching/cs5140/A5/S2.txt`

The first data set S1 has a set of $m = 3{,}000{,}000$ characters, and the second one S2 has $m = 4{,}000{,}000$ characters. The order of the file represents the order of the stream.

1 Streaming Algorithms

A (40 points): Mirsra Gries
S_1 ={'a': 736664, 'b': 436662, 'c': 197649, 'd': 1, 'p': 1, 'v': 1, 't': 1, 'n': 1}
The estimated ratio of a is 0.246
The estimated ratio of b is 0.146
The estimated ratio of c is 0.066
The estimated ratio of d is 0.0
The estimated ratio of p is 0.0
The estimated ratio of v is 0.0
The estimated ratio of t is 0.0
The estimated ratio of n is 0.0

Applying this formula to find the lower bound:

$$f_q - \frac{m}{k} \leq \hat{f}_q \leq f_q$$

Assuming we can round the result to 20%
Might be greater than 20%: b but very close to upperbound
Must be greater than 20%:a

lower ratio bound of a is 0.199855$\approx$ .20
uper ratio bound/actual ratio of a is 0.299855$\approx$ .30
lower ratio bound of b is 0.09985$\approx$ .10
uper ratio bound/actual ratio of b is 0.19985$\approx$ .20

S_2 ={'b': 685133, 'a': 1885833, 'c': 286358, 'f': 1, 'h': 1, 'j': 1, 'o': 1, 'i': 1, 'v': 1}
The estimated ratio of b is 0.171
The estimated ratio of a is 0.471
The estimated ratio of c is 0.072

The estimated ratio of f is 0.0
The estimated ratio of h is 0.0
The estimated ratio of j is 0.0
The estimated ratio of o is 0.0
The estimated ratio of i is 0.0
The estimated ratio of v is 0.0

Might be greater than 20%: b but very close to upper bound
Must be greater than 20%:a

lower ratio bound of b is 0.09985$\approx$ .10
uper ratio bound/actual ratio of b is 0.19985$\approx$ .20
lower ratio bound of a is 0.400025$\approx$ .40
uper ratio bound/actual ratio of a is 0.500025$\approx$ .50
Conclustion: Misra-Gries Algorithm undecounts

```
a= s1 # input stream
k = 10
d = {} # dictionary with key is label and value is counter
length = k - 1
print(len(a))
sub = 0

def decremented():
    for c in list(d):
        if d[c] == 1:
            del d[c]
            #print(d)
        else:
            d[c] -= 1

#Misra Gries
for i in range(0, len(a)):
    if a[i] in d:
        d[a[i]] += 1
        #print(d)
    else:
        if len(d) < length:
              d[a[i]] = 1 # set C[j] =1
        else:
            #print('subtract',d)
            decremented()
            sub += 1
print (d) # dictionary with labels and counters


for l, c in d.items():
  print ('The estimated ratio of ', l , " is " , round(float(c)/len(a),3) )
  count = 0
  for char in a:
      if char == l:
        count += 1
  #print ('lower bound of ',l, ' is ' , count-(len(a)/k))
  print ('lower ratio bound of ',l, ' is ' , (count-(len(a)/k))/len(a))
  print ('uper ratio bound/actual ratio of ',l, ' is ' , count/len(a))
```

B (40 points): Count-Min Sketch
S_1:
a - 959642
b - 659576
c - 420301
The estimated ratio of a is 0.32
The estimated ratio of b is 0.22
The estimated ratio of c is 0.14

The bound for Count Min Sketch is

$$f_q \leq \hat{f}_q \leq f_q + \epsilon F_1$$

Might be greater than 20%: b
Must be greater than 20%:a

lower ratio bound/actual ratio of a is 0.299855$\approx$ .30
lower ratio bound/actual ratio of b is 0.19985$\approx$ .20

S_2:
a - 2039790
b - 839326
c - 440649
The estimated ratio of a is 0.51
The estimated ratio of b is 0.21
The estimated ratio of c is 0.11

Might be greater than 20%: b
Must be greater than 20%:a

lower ratio bound/actual ratio of b is 0.19985$\approx$ .20
lower ratio bound/actual ratio of a is 0.500025$\approx$ .50
Conclustion: Count Min Sketch Algorithm overcounts

C (10 points): Tweets:
Instead of using 1 single character, we can parse into k-grams of words or characters.
For Misra Gries, the label would be each word, therefore this is very large amount of labels and it could be challenging to store.
In Count Min Hash, we just have to hash each words into a hash function, this is more optimal.

D (10 points): Count-Min vs Misra-Gries:
Count-Min is extremely fast as we just compute a hash, and update one value in each of a small number of counters C
Misra-Gries may need to go over $1/\epsilon$ values and decrease them.