Mandatory 2: HyperLogLog

02807 Computational Tools for Data Science

Submission opens:	**08:00 Saturday, October 6, 2018**
Deadline:	**20:00 Sunday, October 21, 2018**
Individual:	**This assignment must be completed individually** (see collaboration policy on course page)

Exercise

This mandatory assignment is about analyzing sales data for a big company X that gets in lots of sales data. For this reason, you are not able to store the original data, but must still be able to give some estimations about the sales made. Also, the data about the sales come from many different systems in the company, and therefore information about the same sale might be reported to you multiple times. This assignment is divided into two parts:

In the first part, you should look up the HyperLogLog on `https://en.wikipedia.org/wiki/HyperLogLog` and understand how it works. Afterwards you should do an implementation of this using the provided template in Python (HyperLogLog.py). We have provided a test script (HyperLogLogTester.py) and two small test files (HHLSampleX.txt). You may need to create your own bigger test files for debugging/testing.

In the second part, you should use your HyperLogLog implementation to solve the problem of storing sales data for company X. In particular, you are asked to estimate the number of sales that have some given attributes. See the file SalesAnalyzer.py for details about the sales data, and what queries can be made about the sales data (all queries can be solved by using (multiple) HyperLogLogs in the right ways). We have provided a test script (SalesAnalyzerTester.py) and a small test file (SASample1.txt). Again, you may need to create your own bigger test files for debugging/testing.

We strongly recommend you start out making your HyperLogLog.py file work correctly, and then afterwards the SalesAnalyzer.py. As always, we recommend you implement one function at a time and make sure it works (not just on our sample test data).

The functions in both classes may be called in any order and multiple times in the tests.

Submission You must submit your solution on CodeJudge **before** the deadline (any submissions made after the deadline will **not** be counted). Your latest submission before the deadline will be used for grading thus you **must** ensure your best solution is uploaded as the latest. Submission on CodeJudge will not be available before the submission opens time stated in the top.

Grading In order to obtain full points for this assignment, you must implement all the described functions in a correct and efficient way. From submission opens until the deadline you can submit your code on CodeJudge and see how well it scores. An entirely correct solution gets 100 points. Partially correct/efficient solutions will get a score between 0 and 100. On CodeJudge you can see the results of all the tests, but all except the already provided sample test data will be hidden.

Correctness and Efficiency. The main focus of this assignment is to do a correct implementation of HyperLogLog that is able to estimate numbers close to the correct ones. In all tests for this assignment there is a 10MB memory limit (be aware importing packages may take up your memory, so only use strictly needed packages).

The estimates given by the different functions you are to implement should have an relative error of at most 20% or an absolute error of at most 4.000[1], otherwise the answer is considered wrong.

[1] Ie. if the correct answer is 1.000, any answer in the range 0-5.000 will be considered correct, and if the correct answer is 1.000.000 any answer in the range 800.000-1.200.000 will be considered correct

The score you obtain in this mandatory assignment will count towards your final grade.

Competition If you are up for an extra challenge, there will be a competition among those interested to make the fastest possible solution. This is completely voluntarily, and will **not** count towards your final grade – instead there will be a small symbolic prize for the best solution.

The problem in the competition is exactly like the second part of the mandatory part and the tests will be of the same type, so everybody can join the competition with their solution without extra effort.

In this competition, we will mainly be competing on the precision of the estimates. For each estimate made during a test, we will calculate the value $100 - \min(\frac{500 \cdot |c-e|}{c}, \frac{|c-e|}{40})$ where e is your estimate and c is the correct value. If this value is negative for any estimation made during the test, we will consider the test as failed and it will get a score of 0, otherwise the score of the test will be the rounded average of this value for all estimations in the test. Tests that fail for some reason (for instance due runtime or time limit errors) will get a score of 0. The winner will be the one with the highest sum of scores.

You are free to use any approach you like for the competition (ie. you do not even have to use HyperLogLog). Just remember there is a memory limit.

To join the competition, simply also submit your best solution to the "Competition" exercise on CodeJudge (you must still submit to the mandatory exercise as well). A live scoreboard will show the current standings on CodeJudge.

Hints

- See Figure 1 in `https://storage.googleapis.com/pub-tools-public-publication-data/pdf/40671.pdf` for how to decide the value of α_m (you do not need the small range/large range corrections presented in the pseudo code in the Figure in order to pass the mandatory assignment)
- See `https://odino.org/my-favorite-data-structure-hyperloglog/` for a story that tries to describe the intuition behind HyperLogLog (decide for yourself if you find it helpful or not)
- The 32-bit hash function in the mmh3 package will be a fine hash function for this assignment (ie. mmh3.hash(...)).
- See `https://docs.python.org/3.6/library/stdtypes.html` for bit manipulation in Python (in particular you may find the bit_length() function relevant)