Midterm – 200 points
Due April 2nd, 11:55PM, via NYU Brightspace

SUBMIT YOUR CODE/SOLUTIONS IN A ZIP FILE

Note: There are 3 questions adding up to 225 points. 200 points is a perfect score.

1. Random Sampling a *data stream* – 75 points

The Problem Statement:

Your data is a stream of items of ***unknown length*** that we can only iterate over once. Also, the data is expected to be very large, so while you are developing your program, you'd like to work on a *statistically accurate* representative sample.

You would need to implement an algorithm that randomly chooses an item from the data stream ***such that each item is equally likely to be selected.***

The Algorithm:

The algorithm for this problem is the Reservoir Sampling algorithm. http://en.wikipedia.org/wiki/Reservoir_sampling.

An good simpler explanation and a ***Python-only*** implementation is shown here: https://towardsdatascience.com/the-5-sampling-algorithms-every-data-scientist-need-to-know-43c7bc11d17c (if you hit paywall for this link, retry it on an incognito window).

The Data: data_Q1_2019.zip

This dataset contains the actual daily SMART logs for all hard drives used in a data center during the first quarter of 2019. Note that over the course of the three months, some drives will fail and new one will come into use. There are over 900k entries in this dataset.
SMART, https://en.wikipedia.org/wiki/Self-Monitoring,_Analysis_and_Reporting_Technology

TODO: Create a random subset of 50k entries of this data using Reservoir Sampling. (there are about 900K entries)

Sample size = k = 50,000 = 50k

1. Implement Reservoir Sampling in Hadoop MapReduce - 50 points
2. Implement Reservoir Sampling in Spark - 25 points

2. Probabilistic Set Membership – 75 points

Problem Statement

You want to have a really fast and quick method to determine if an entity is a member of a set – test for set membership. The set of interest (membership) would have a small number of items k, but the candidate pool of items to test, n, can be in the millions or billions.

n << n

The Algorithm:

A Bloom filter is an efficient probabilistic data structure constructed from a set of values and used for fast set membership tests. A Bloom filter can tell you if an arbitrary element being tested **might** be in the set, or **definitely not** in the set.

That is false positives (FP) are allowed, but false negatives (FN) are not.

Bloom Filter: https://en.wikipedia.org/wiki/Bloom_filter

The data structure is a bit array, onto which elements are mapped using hash functions. The mapping sets some bits to 1 leaving the rest as 0's. The size of the bit array is determined by how much false positives you are willing to tolerate, so most implementations will accept a false positive rate (FPR) parameter in the constructor (typical value is 1%).

The Data: data_Q1_2019.zip

TODO (in Spark):

1. Split the dataset by month: January, February and March. – 10 points
2. Display the total actual hard drives active during each month - 5 points
3. Define the membership set as those hard drives (**model name)** with zero error rate during January and February. Train a bloom filter with that set – 40 points
4. Test the March drives for set membership – 20 points

NOTE:

Use PyBloom. https://pypi.org/project/pybloom/
PyBloom is already installed in the JupyuterHub environment for this class.

3. Duplicate Detection – 75 points

Problem Statement

You teach a course at NYU and trust your students to learn the material by doing original work. However, you find many cases of plagiarism and copying. You need an easy method to detect copying and plagiarism so you can fail those students.

The Algorithm:

Minhash/LSH is an algorithm based on cryptographic hashes that computes similarity between a pair of entities. It is heavily used in Web Search and product similarity. Details of the algorithm can be found in Chapter 3 of *Mining Massive* Datasets.
http://infolab.stanford.edu/~ullman/mmds/ch3.pdf

The Data: articles1.csv.zip

TODO (in Spark): 75 points

Use the Minhash/LSH algorithm to detect ***10 most similar*** entries in articles1.csv to a reference article. The reference article is in the dataset, identified as Article ID 69716, “California lifted its mandatory water restrictions - that could be a huge mistake”. The reference article should be excluded from the solution.

The Minhash/LSH algorithm relies on the concept of distances to define similarity. For this exercise, **use Jaccard** similarity.

NOTE:

For this problem, you can use the Python version in datasketch ,
http://ekzhu.com/datasketch/minhash.html

Datasketch is already installed in the JupyterHub environment for this class.

Spark ML has Minhash/LSH library. You could use that, but we have not studied the ML package yet.