Fall 2018: Introduction to Data Science

GIRI NARASIMHAN, SCIS, FIU

The Stream Model

- Data arrives in a stream
- Data is arriving rapidly
- Data cannot be stored in local storage, but in archival storage
- Archival storage, if any, is too large and cannot be accessed quickly
- Archival storage cannot be searched quickly
- If stream data is not processed immediately, then it is lost
- Decisions have to be made based on the data
- Quick approximate answer is often better than slow exact answer

Examples

- Wall street stock market data
- Satellite image data
- Internet and web traffic data
- Sensor data
  - 4-byte data every 0.1 sec = 3.5 MB/day
  - 1 million sensors in the ocean corresponds to one e
  - 40 MB every sec

Queries

- Alert when temperature is above 25 degrees
- Sliding window concept
  - Maximum temperature for period X
  - Alert when average for X is above 25 degrees
  - Number of unique elements for X

Standard Trick: Random Sampling

- **Random Sampling**: Pick a random integer from [0 .. N-1] and if 0, process the stream data, else ignore it.
  - Samples 1/N items
- It artificially **slows down the stream** to manageable levels

Sampling Woes

- **Stream**: Tuples (**user**, **query**, **time**); **Sampling**: 1 in 10
  - Each user has 1/10 of their queries processed
- **Query**: Fraction of typical user's queries repeated over last month
- **Correct Answer**: Suppose user has s unique queries and d queries twice and NO queries more than twice in the last month; Answer = d/(s+d)
- **Problem**: Reported fraction would be wrong
  - In the sampled stream, s/10 are unique queries and d/100 queries appear twice
  - The remainder of the queries that should appear twice will appear once 18d/100
  - We will report d/(10s + 19d) [d/100 twice and s/10 + 18d/100 once

Improved Solution for Sampling Woes

- Problem is that we are picking 1/10 of the queries
- We need to pick 1/10 of the users and pick all their queries
- If we can store 1/10 of the users, then for every query we can decide either to process or not
- **Improved Solution**: Hash user ID (actually, IP address) to 0 … 9
  - Pick only those that hash to 0
- **Sampling Question**: How to sample at rate of 1/70?
- **Sampling Question**: How to sample at rate of 23/70?

Sampling

- Sampling can be applied if the filtering test is easy (e.g., hash value = 0? Temperature > 22 degrees?)
- Sampling is harder if it involves a lookup (e.g., has this query been asked before by this user? Is this user among the top 10% of the frequent users list?)
- Other techniques are available for filtering
  - **Bloom Filters**

Example: Bloom Filters for Spam

- **White lists**: allowed email addresses
  - Assume we have **1 Billion** allowed email addresses
  - Assume black list is much larger than white list
  - If each email address is 20 bytes, this takes 20 GB to store
- **Bloom Filters**: store white lists as bit hash arrays
  - Every email address is hashed and a 1 is stored in the location if it is in white list
  - In 1 GB, we can store hash array of size 8 Billion
- Strict White Lists: use bloom filters and then verify with real white list
- Stricter White List: use cascade of bloom filters

Bloom Filters: Test for Membership

- Array of n bits, initially all 0's
- Collection of k hash functions. Each hash func maps a key to n buckets
- Given key K, compute K hash values and
  - Check that each location in bit array is a 1
  - Even if one is 0, then it fails the test

False Positive Rate

$(1\text{-}h)^{1/h} = e$ f
small h

- Assume we have **x** targets and **y** darts
- Prob a dart will hit a specific target = 1/x
- Prob a dart does not hit a specific target = 1 – (1/x) = (x-1)/x
- Prob that y darts miss a specific target = $((x-1)/x)^y$
- Prob that y darts miss a specific target = $e^{-y/x}$
- Let x = 8B; y = 1B; Then prob of missing a target = $e^{-1/8}$
- Prob of hitting a target = false positive rate = $1 - e^{-1/8} = 0.1175$
- If k = 2, the prob becomes $(1 - e^{-1/4})^2 = 0.0493$

False Positive Rate

- Let n = bit array length = 8B
- Let m = # of members = 1B
- Let k = # of hash functions = 1
- Prob that a white list email hashes to a location = 10-9

Counting distinct elements

- How many unique users in a give period?
- How many users (IP addresses) visited a webpage?
  - Each IP address is 4 bytes = 32 bits
  - 4 billion IP addresses are possible = 16 GB
  - If we need this for each webpage and there are thousands, then we cannot store in memory

Flajolet-Martin Algorithm

- For each element obtain a sufficiently long hash
  - Has to be more possible results of hash than elements in the universal set
  - Example, use 64 bits ($2^{64} \sim 10^{19}$) to hash URLs (4 Billion)
  - High prob that different elements get different hash values
  - Some fraction of these hash values will be "unusual"
- We will focus on the ones that have r 0s at the end of its hash value
  - Prob of hash value to end in r 0s is 2-r
  - Prob that m unique items have has values that don't end in r 0s is $(1-2^{-r})^m = e^{-m2^{-r}}$

Summary

- Look at the probability = $e^{-m2^{-r}}$
- If m is much larger than 2^r, then prob approaches 1
- If m is much smaller than 2^r, then prob approaches 0
- Thus 2^R is a good choice, where R is the largest tail of 0s

Moments

- i-th Momemt
- Zeroth Moment
- First Moment
- Average = ?
- Variance = ?

$$\frac{1}{m}\sum_{s=1}^{m}\left(f_s - \frac{n}{m}\right)^2 = \frac{1}{m}\sum_{s=1}^{m}\left(f_s^2 - 2\frac{n}{m}f_s + \left(\frac{n}{m}\right)^2\right) = \left(\frac{1}{m}\sum_{s=1}^{m} f_s^2\right) - \frac{n^2}{m^2}$$