CSE 312

Foundations of Computing II

20: Counting Distinct Elements

www.slido.com/2226110

Data mining – Stream Model

- In many data mining situations, data often not known ahead of time.
  - Examples: Google queries, Twitter or Facebook status updates, YouTube video views
- Think of the data as an <u>infinite stream</u>
- Input elements (e.g. Google queries) enter/arrive one at a time.
  - We cannot possibly store the stream.

Question: How do we make critical calculations about the data stream using a limited amount of memory?

Stream Model – Problem Setup

Input: sequence (aka. "stream") of N elements $x_1, x_2, \ldots, x_N$ from a known universe U (e.g., 8-byte integers).

Goal: perform a computation on the input, in a single left to right pass, where:

- Elements processed in real time
- Can't store the full data $\Rightarrow$ use minimal amount of storage while maintaining working "summary"

What can we compute?

32, 12, 14, 32, 7, 12, 32, 7, 32, 12, 4

Some functions are easy:

- Min
- Max
- Sum
- Average

Today: Counting distinct elements

32, 12, 14, 32, 7, 12, 32, 7, 32, 12, 4

Application

You are the content manager at YouTube, and you are trying to figure out the **distinct** view count for a video. How do we do that?

Note: A person can view their favorite videos several times, but they only count as 1 **distinct** view!

Other applications

- IP packet streams: How many distinct IP addresses or IP flows (source+destination IP, port, protocol)
  - Anomaly detection, traffic monitoring
- Search: How many distinct search queries on Google on a certain topic yesterday
- Web services: how many distinct users (cookies) searched/browsed a certain term/item
  - Advertising, marketing trends, etc.

Counting distinct elements

32, 12, 14, 32, 7, 12, 32, 7, 32, 12, 4

N = # of IDs in the stream = 11, m = # of distinct IDs in the stream = 5

Want to compute number of **distinct** IDs in the stream.

- *Naïve solution:* *As the data stream comes in, store all distinct IDs in a hash table.*
- *Space requirement:* $\Omega(m)$

YouTube Scenario: m is huge!

Counting distinct elements

32, 12, 14, 32, 7, 12, 32, 7, 32, 12, 4

N $=$ # of IDs in the stream = 11, m $=$ # of distinct IDs in the stream = 5

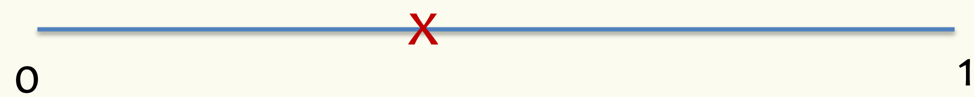
Want to compute number of **distinct** IDs in the stream.

How to do this <u>without</u> storing all the elements?

Detour – I.I.D. Uniforms

If $Y_1, \cdots, Y_m \sim \mathrm{Unif}(0,1)$ (i.i.d.) where do we expect the points to end up?

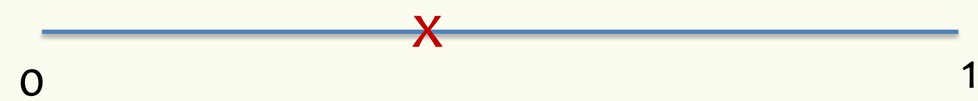
$m = 1$



Detour – I.I.D. Uniforms

If $Y_1, \cdots, Y_m \sim \mathrm{Unif}(0,1)$ (i.i.d.) where do we expect the points to end up?

$m = 1$



$m = 2$

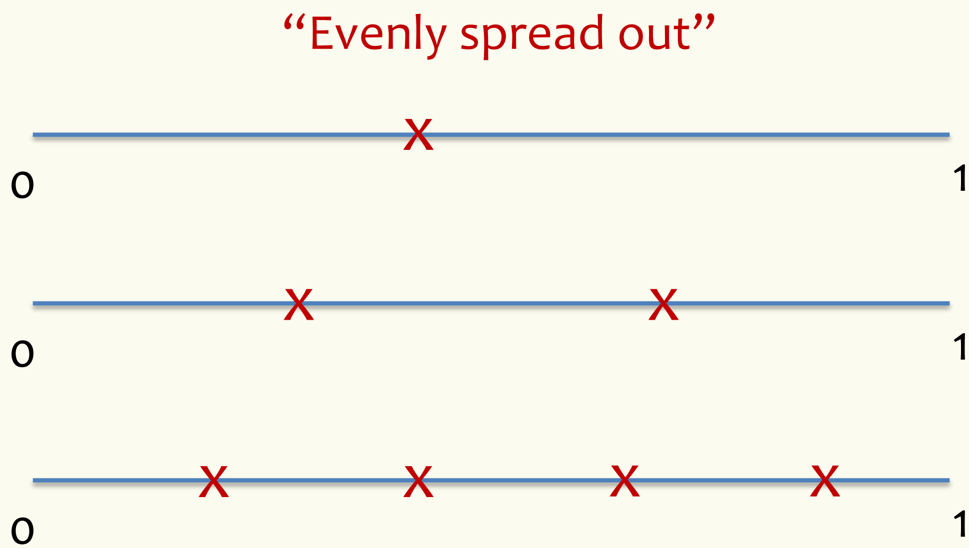
Detour – I.I.D. Uniforms

If $Y_1, \cdots, Y_m \sim \mathrm{Unif}(0,1)$ (i.i.d.) where do we expect the points to end up?

$m = 1$

$m = 2$

$m = 4$



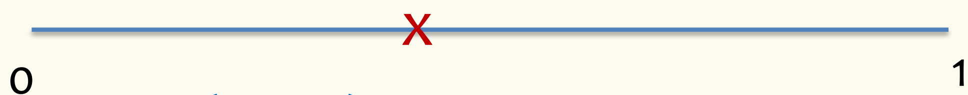
Detour – Min of I.I.D. Uniforms

If $Y_1, \cdots, Y_m \sim \mathrm{Unif}(0,1)$ (iid) where do we expect the points to end up?

In general, $\mathbb{E}[\min(Y_1, \cdots, Y_m)] = \frac{1}{m+1}$

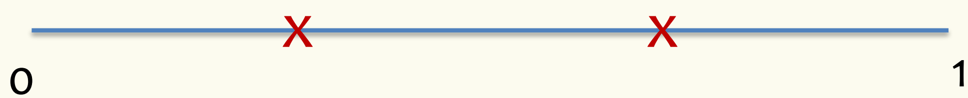
$\mathbb{E}[\min(Y_1)] =$

$m = 1$



$\mathbb{E}[\min(Y_1, Y_2)] =$

$m = 2$



$\mathbb{E}[\min(Y_1, \cdots, Y_4)] =$

$m = 4$

Detour – Min of I.I.D. Uniforms

If $Y_1, \cdots, Y_m \sim \mathrm{Unif}(0,1)$ (iid) where do we expect the points to end up?

In general, $\mathbb{E}[\min(Y_1, \cdots, Y_m)] = \frac{1}{m+1}$

What is some intuition for this?

Detour – Min of I.I.D. Uniforms

If $Y_1, \cdots, Y_m \sim \mathrm{Unif}(0,1)$ (i.i.d.) where do we expect the points to end up?

e.g., what is $\mathbb{E}[\min\{Y_1, \cdots, Y_m\}]$?

CDF: Observe that $\min\{Y_1, \cdots, Y_m\} \geq y$ if and only if $Y_1 \geq y, \dots, Y_m \geq y$

$$\begin{aligned} \underset{y \in [0,1]}{P(\min\{Y_1, \cdots, Y_m\} \geq y)} &= P(Y_1 \geq y, \dots, Y_m \geq y) \\ &= P(Y_1 \geq y) \cdots P(Y_m \geq y) \qquad \text{(Independence)} \\ &= (1-y)^m \end{aligned}$$

$$\Rightarrow P(\min\{Y_1, \cdots, Y_m\} \leq y) = 1 - (1-y)^m$$

Detour – Min of I.I.D. Uniforms

Useful fact. For any random variable Y taking non-negative values

$$\mathbb{E}[Y] = \int_0^\infty P(Y \geq y) \mathrm{d}y$$

Proof

$$\mathbb{E}[Y] = \int_0^\infty x \cdot f_Y(x)\,\mathrm{d}x = \int_0^\infty \left(\int_0^x 1\,\mathrm{d}y\right) \cdot f_Y(x)\,\mathrm{d}x = \int_0^\infty \int_0^x f_Y(x)\,\mathrm{d}y\,\mathrm{d}x$$

$$= \int_0^\infty \int_y^\infty f_Y(x)\,\mathrm{d}x\,\mathrm{d}y = \int_0^\infty P(Y \geq y)\,\mathrm{d}y$$

Detour – Min of I.I.D. Uniforms

$Y_1, \cdots, Y_m \sim \mathrm{Unif}(0,1)$ (i.i.d.)

$Y = \min\{Y_1, \cdots, Y_m\}$

Useful fact. For any random variable Y taking non-negative values

$$\mathbb{E}[Y] = \int_0^\infty P(Y \geq y) \mathrm{d}y$$

$$\mathbb{E}[Y] = \int_0^\infty P(Y \geq y) \mathrm{d}y = \int_0^1 (1-y)^m \mathrm{d}y$$

$$= -\frac{1}{m+1}(1-y)^{m+1}\Big|_0^1 = 0 - \left(-\frac{1}{m+1}\right) = \boxed{\frac{1}{m+1}}$$

Detour – Min of I.I.D. Uniforms

If $Y_1, \cdots, Y_m \sim \mathrm{Unif}(0,1)$ (iid) where do we expect the points to end up?

In general, $\mathbb{E}[\min(Y_1, \cdots, Y_m)] = \frac{1}{m+1}$

$$\mathbb{E}[\min(Y_1)] = \frac{1}{1+1} = \frac{1}{2}$$

$m = 1$

$$\mathbb{E}[\min(Y_1, Y_2)] = \frac{1}{2+1} = \frac{1}{3}$$

$m = 2$

$$\mathbb{E}[\min(Y_1, \cdots, Y_4)] = \frac{1}{4+1} = \frac{1}{5}$$

$m = 4$

Back to counting distinct elements

32, 12, 14, 32, 7, 12, 32, 7, 32, 12, 4

N = # of IDs in the stream = 11, m = # of distinct IDs in the stream = 5

Want to compute number of **distinct** IDs in the stream.

How to do this <u>without</u> storing all the elements?

Distinct Elements – Hashing into $[0, 1]$

Hash function $h: U \to [0,1]$
Assumption: For all $x \in U$, $h(x) \sim \mathrm{Unif}(0,1)$ and mutually independent

$$
\begin{array}{cccccccc}
32, & 12, & 14, & 32, & 7, & 12, & 32, & 7 \\
\downarrow & \downarrow & \downarrow & \downarrow & \downarrow & \downarrow & \downarrow & \downarrow \\
h(32), & h(12), & h(14), & h(32), & h(7), & h(12), & h(32), & h(7)
\end{array}
$$

Distinct Elements – Hashing into $[0, 1]$

Hash function $h: U \to [0,1]$
Assumption: For all $x \in U$, $h(x) \sim \mathrm{Unif}(0,1)$ and mutually independent

32, 12, 14, 32, 7, 12, 32, 7

$\downarrow$

h(32), h(12), h(14), h(32), h(7), h(12), h(32), h(7)

4 distinct elements

$\rightarrow$ 4 i.i.d. RVs $h(32), h(12), h(14), h(7) \sim \mathrm{Unif}(0,1)$

$\rightarrow \mathbb{E}[\min\{h(32), h(12), h(14), h(7)\}] = \frac{1}{5+1} = \frac{1}{6}$

Distinct Elements – Hashing into $[0, 1]$

Hash function $h: U \to [0,1]$
Assumption: For all $x \in U$, $h(x) \sim \text{Unif}(0,1)$ and mutually independent

$x_1, x_2, \dots, x_N$ contains m distinct elements

⬇

$h(x_1), h(x_2), \dots, h(x_N)$ contains m i.i.d. rvs $\sim \text{Unif}(0,1)$

and $N - m$ repeats

⬇

$$\mathbb{E}[\min\{h(x_1), \dots, h(x_N)\}] = \frac{1}{m+1}$$

A super duper clever idea!!!!

$$\mathbb{E}[\min\{h(x_1), \dots, h(x_N)\}] = \frac{1}{m+1}$$

So $m = \frac{1}{\mathbb{E}[\min\{h(x_1),\dots,h(x_N)\}]} - 1$

What if $\min\{h(x_1), \dots, h(x_N)\}$ is $\approx \mathbb{E}[\min\{h(x_1), \dots, h(x_N)\}]$?

The MinHash Algorithm – Idea

$$m = \frac{1}{\mathbb{E}[\min\{h(x_1), \dots, h(x_N)\}]} - 1$$

1. Compute $\mathrm{val} = \min\{h(x_1), \dots, h(x_N)\}$
2. Assume that $\mathrm{val} \approx \mathbb{E}[\min\{h(x_1), \dots, h(x_N)\}]$
3. Output as estimate for m: $\quad \mathrm{round}\left(\frac{1}{\mathrm{val}} - 1\right)$

The MinHash Algorithm – Implementation

Algorithm **`MinHash`**$(x_1, x_2, \ldots, x_N)$

$\text{val} \leftarrow \infty$

for $i = 1$ **to** N **do**

$\quad \text{val} \leftarrow \min\{\text{val}, h(x_i)\}$

return $\text{round}\left(\frac{1}{\text{val}} - 1\right)$

Memory cost = just remember val (with sufficient precision)

1. Compute $\text{val} = \min\{h(x_1), \dots, h(x_N)\}$
2. Assume that $\text{val} \approx \mathbb{E}[\min\{h(x_1), \dots, h(x_N)\}]$
3. Output $\text{round}\left(\frac{1}{\text{val}} - 1\right)$

MinHash Example

Stream: 13, 25, 19, 25, 19, 19

Hashes: 0.51, 0.26, 0.79, 0.26, 0.79, 0.79

What does MinHash return?

Poll: **www.slido.com/2226110**

a. 1
b. 3
c. 5
d. No idea

MinHash Example II

Stream: 11, 34, 89, 11, 89, 23

Hashes: 0.5, 0.21, 0.94, 0.5, 0.94, 0.1

Output is $\frac{1}{0.1} - 1 = 9$

Clearly, not a very good answer!

Not unlikely: $P(h(x) < 0.1) = 0.1$

The MinHash Algorithm – Problem

Algorithm **`MinHash`**$(x_1, x_2, \dots, x_N)$

$\mathrm{val} \leftarrow \infty$

for $i = 1$ **to** N **do**

$\mathrm{val} \leftarrow \min\{\mathrm{val}, h(x_i)\}$

return $\mathrm{round}\left(\frac{1}{\mathrm{val}} - 1\right)$

But val is not $\mathbb{E}[\mathrm{val}]$!
How far is val from $\mathbb{E}[\mathrm{val}]$?

$$\mathrm{Var}(\mathrm{val}) \approx \frac{1}{(m+1)^2}$$

$\mathrm{val} = \min\{h(x_1), \dots, h(x_N)\}$

$\mathbb{E}[\mathrm{val}] = \frac{1}{m+1}$

How can we reduce the variance?

Idea: Repetition to reduce variance!

Use k **independent** hash functions $h^1, h^2, \cdots h^k$

How can we reduce the variance?

Idea: Repetition to reduce variance!

Use k **independent** hash functions $h^1, h^2, \cdots h^k$

Algorithm **`MinHash`**$(x_1, x_2, \dots, x_N)$

$\mathrm{val}_1, \dots, \mathrm{val}_\mathrm{k} \leftarrow \infty$

for $i = 1$ **to** N **do**

$\quad \mathrm{val}_1 \leftarrow \min\{\mathrm{val}_1, h^1(x_i)\}, \dots, \mathrm{val}_\mathrm{k} \leftarrow \min\{\mathrm{val}_k, h^k(x_i)\}$

$\mathrm{val} \leftarrow \frac{1}{k}\sum_{i=1}^{k} \mathrm{val}_\mathrm{i}$

return $\mathrm{round}\left(\frac{1}{\mathrm{val}} - 1\right)$

$$\boxed{\mathrm{Var}(\mathrm{val}) = \frac{1}{k}\frac{1}{(m+1)^2}}$$

MinHash and Estimating # of Distinct Elements in Practice

- MinHash in practice:
  - One also stores the element that has the minimum hash value for each of the k hash functions
    - Then, just given separate MinHashes for sets A and B, can also estimate
      - what fraction of $A \cup B$ is in $A \cap B$; i.e., how similar A and B are