CSE 312

Foundations of Computing II

22: Wrap up MLE; Counting Distinct Elements

Agenda

- Recap MLE
- Unbiased and Consistent Estimators
- Distinct Elements Application

Estimation

Distribution
$P(x;\theta)$

θ = unknown parameter

Independent samples
$X_1, \ldots, X_n$
from $P(x;\theta)$

Estimation Algorithm

Parameter estimate

$\hat{\theta}_n\,(X_1, \ldots, X_n)$

Likelihood of Different Observations

(Discrete case)

Definition. The **likelihood** of independent observations $x_1, \dots, x_n$ is

$$\mathcal{L}(x_1, x_2, \dots, x_n\,; \theta) = \prod_{i=1}^{n} P(x_i; \theta)$$

Maximum Likelihood Estimation (MLE). Given data $x_1, \dots, x_n$, find $\hat{\theta}$ such that $\mathcal{L}\left(x_1, x_2, \dots, x_n\,; \hat{\theta}\right)$ is maximized!

$$\hat{\theta} = \underset{\theta}{\operatorname{argmax}}\ \mathcal{L}(x_1, x_2, \dots, x_n\,; \theta)$$

General Recipe

1. **Input** Given n i.i.d. samples $x_1, \ldots, x_n$ from parametric model with parameter θ.
2. **Likelihood** Define your likelihood $\mathcal{L}(x_1, \ldots, x_n ; \theta)$.
   - For discrete $\mathcal{L}(x_1, \ldots, x_n ; \theta) = \prod_{i=1}^{n} P(x_i ; \theta)$
   - For continuous $\mathcal{L}(x_1, \ldots, x_n ; \theta) = \prod_{i=1}^{n} f(x_i ; \theta)$
3. **Log** Compute $\ln \mathcal{L}(x_1, \ldots, x_n ; \theta)$
4. **Differentiate** Compute $\frac{\partial}{\partial \theta} \ln \mathcal{L}(x_1, \ldots, x_n ; \theta)$
5. **Solve for** $\hat{\theta}$ by setting derivative to 0 and solving for max.

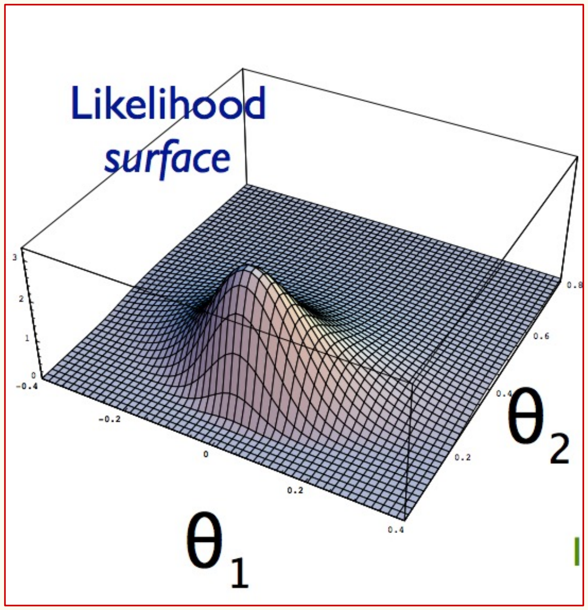
Generally, you need to do a second derivative test to verify it is a maximum, but we won't ask you to do that in CSE 312.

$$\ln(ab) = \ln(a) + \ln(b)$$
$$\ln(a/b) = \ln(a) - \ln(b)$$
$$\ln(a^b) = b \cdot \ln(a)$$

Two-parameter optimization

Normal outcomes $x_1, \ldots, x_n$

Goal: estimate θ_μ = expectation and θ_{σ^2} = variance



$$\mathcal{L}(x_1, \ldots, x_n \,; \theta_\mu, \theta_{\sigma^2}) = \left(\frac{1}{\sqrt{2\pi\theta_{\sigma^2}}}\right)^n \prod_{i=1}^{n} e^{-\frac{(x_i - \theta_\mu)^2}{2\theta_{\sigma^2}}}$$

$$\ln \mathcal{L}(x_1, \ldots, x_n \,; \theta_\mu, \theta_{\sigma^2}) = -n\frac{\ln(2\pi\,\theta_{\sigma^2})}{2} - \sum_{i=1}^{n} \frac{(x_i - \theta_\mu)^2}{2\theta_{\sigma^2}}$$

Likelihood – Continuous Case

Definition. The **likelihood** of independent observations $x_1, \ldots, x_n$ is

$$\mathcal{L}(x_1, \ldots, x_n\,; \theta) = \prod_{i=1}^{n} f(x_i; \theta)$$

Normal outcomes $x_1, \ldots, x_n$

$$\hat{\theta}_\mu = \frac{\sum_i^n x_i}{n}$$

MLE estimator for **expectation**

$$\hat{\theta}_{\sigma^2} = \frac{1}{n}\sum_{i=1}^{n}\left(x_i - \hat{\theta}_\mu\right)^2$$

MLE estimator for **variance**

Agenda

- Recap MLE
- Unbiased and Consistent Estimators
- Distinct Elements Application

When is an estimator good?

Parameter estimate

Distribution $P(x;\theta)$ → Independent samples $X_1, \dots, X_n$ from $P(x;\theta)$ → **Estimation Algorithm** → $\hat{\theta}_n\,(X_1, \dots, X_n)$

θ = unknown parameter

Definition. An estimator of parameter θ is an **unbiased estimator** if

$$\mathbb{E}[\hat{\theta}_n] = \theta.$$

Note: This expectation is over the samples $X_1, \dots, X_n$

Three samples from $U(0, \theta)$

Example – Coin Flips

Recall: $\hat{\theta}_\mu = \frac{n_H}{n}$

Coin-flip outcomes $x_1, \dots, x_n$, with n_H heads, n_T tails

Fact. $\hat{\theta}_\mu$ is unbiased

i.e., $\mathbb{E}[\hat{\theta}_\mu] = p$, where p is the probability that the coin turns out head.

Why?

Because $\mathbb{E}[n_H] = np$ when p is the true probability of heads.

Consistent Estimators & MLE

Distribution $P(x;\theta)$ → Independent samples $X_1, \dots, X_n$ from $P(x;\theta)$ → **Estimation Algorithm** → $\hat{\theta}_n$

Parameter estimate

θ = unknown parameter

Definition. An estimator is **unbiased** if $\mathbb{E}[\hat{\theta}_n] = \theta$ for all $n \geq 1$.

Definition. An estimator is **consistent** if $\lim_{n\to\infty} \mathbb{E}[\hat{\theta}_n] = \theta$.

Theorem. MLE estimators are consistent.

(But not necessarily unbiased)

Example – Consistency

Normal outcomes $X_1, \dots, X_n$ i.i.d. according to $\mathcal{N}(\mu, \sigma^2)$ Assume: $\sigma^2 > 0$

$$\widehat{\Theta}_{\sigma^2} = \frac{1}{n}\sum_{i=1}^{n}\left(X_i - \widehat{\Theta}_{\mu}\right)^2$$

Population variance – Biased!

$\widehat{\Theta}_{\sigma^2}$ is "consistent"

Example – Consistency

Normal outcomes $X_1, \ldots, X_n$ i.i.d. according to $\mathcal{N}(\mu, \sigma^2)$ Assume: $\sigma^2 > 0$

$$\widehat{\Theta}_{\sigma^2} = \frac{1}{n}\sum_{i=1}^{n}\left(X_i - \widehat{\Theta}_{\mu}\right)^2$$

Population variance – <u>Biased!</u>

$$S_n^2 = \frac{1}{n-1}\sum_{i=1}^{n}\left(X_i - \widehat{\Theta}_{\mu}\right)^2$$

Sample variance – <u>Unbiased!</u>

$\widehat{\Theta}_{\sigma^2}$ converges to same value as S_n^2, i.e., σ^2, as $n \to \infty$.

$\widehat{\Theta}_{\sigma^2}$ is "consistent"

So what do we want?

- When statisticians are estimating a variance from a sample, they usually divide by $n-1$ instead of n.

- They and we not only want good estimators (unbiased, consistent)
  - They/we also want **confidence bounds**
    - Upper bounds on the probability that these estimators are far the truth about the underlying distributions
  - Confidence bounds are just like what we wanted for our polling problems, but CLT is usually not the only way or best way to get them (unless the variance is known)

Agenda

- Recap MLE
- Unbiased and Consistent Estimators
- Distinct Elements Application

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 ⇒ 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:

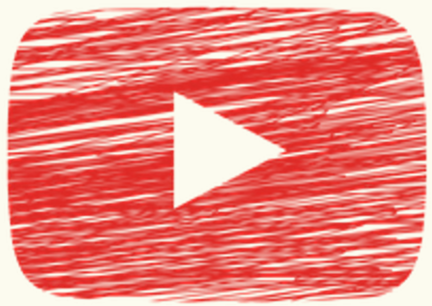
- Min
- Max
- Sum
- Average

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

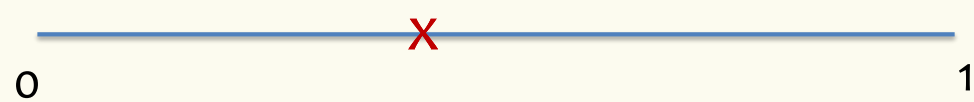
Want to compute number of **distinct** IDs in the stream.

How to do this without storing all the elements?

Detour – I.I.D. Uniforms

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

$m = 1$



Detour – I.I.D. Uniforms

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

$m = 1$

$m = 2$

Detour – I.I.D. Uniforms

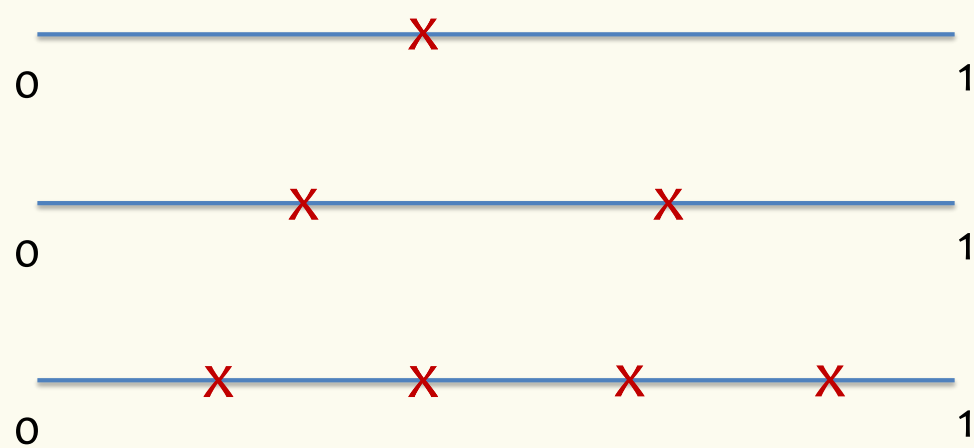
If $Y_1, \cdots, Y_m \sim \mathrm{Unif}(0,1)$ (i.i.d.) where do we expect the points to end up?

"Evenly spread out"

$m = 1$

$m = 2$

$m = 4$



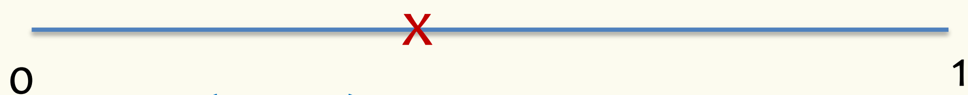
Detour – Min of I.I.D. Uniforms

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

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

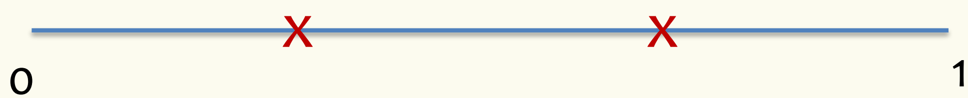
$\mathbb{E}[\min(Y_1)] =$

$m = 1$



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

$m = 2$



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

$m = 4$

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 $Y_1 \geq y, \dots, Y_m \geq y$

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

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

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

$$f_Y(y) = \frac{d}{dy} F_Y(y) = m(1-y)^{m-1}.$$

$$\mathbb{E}[Y] = \int_0^1 y\, f_Y(y)\, \mathrm{d}y = \int_0^1 y\, m(1-y)^{m-1} \mathrm{d}y = \frac{1}{m+1}$$

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

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

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

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

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

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

$\downarrow \quad \downarrow \quad \downarrow \quad \downarrow \quad \downarrow \quad \downarrow \quad \downarrow \quad \downarrow$

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

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

Hash function $h: U \rightarrow [0,1]$
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)

M=4 distinct elements

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

$\rightarrow \mathbb{E}[\min\{h(32), h(12), h(14), h(7)\}] = \frac{1}{4+1} = \frac{1}{5}$

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

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

⬇

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

and $N - m$ repeats

⬇

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

A super duper clever idea!!!!

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

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

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

The MinHash Algorithm – Idea

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

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

The MinHash Algorithm – Implementation

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

$\text{val} \leftarrow \infty$

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

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

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?

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**$(x_1, x_2, \ldots, x_N)$

$\text{val} \leftarrow \infty$

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

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

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

Problem: val is not $\mathbb{E}[\text{val}]$!
How far is val from $\mathbb{E}[\text{val}]$?

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

$\text{val} = \min\{h(x_1), \ldots, h(x_N)\}$

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

How can we reduce the variance?

Idea: Repetition to reduce variance!

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

$$\mathrm{val}_1 = \min\{h^1(x_1), \dots, h^1(x_N)\}$$
$$\mathrm{val}_2 = \min\{h^2(x_1), \dots, h^2(x_N)\}$$
$$\dots$$
$$\mathrm{val}_\mathrm{k} = \min\{h^k(x_1), \dots, h^k(x_N)\}$$

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

Output as estimate

for m: $\mathrm{round}\left(\frac{1}{\mathrm{val}} - 1\right)$

How can we reduce the variance?

Idea: Repetition to reduce variance!

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

Algorithm $\texttt{MinHash}(x_1, x_2, \ldots, x_N)$

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

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

  for $j = 1$ **to** k **do** $\mathrm{val}_j \leftarrow \min\{\mathrm{val}_j, h^j(x_i)\}$

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

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

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