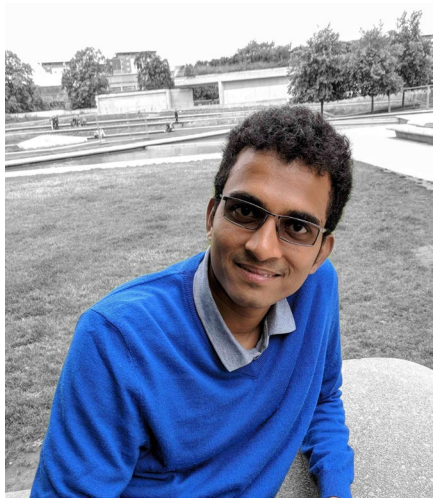
On Differentially Private Counting on Trees

Kewen Wu (UC Berkeley)

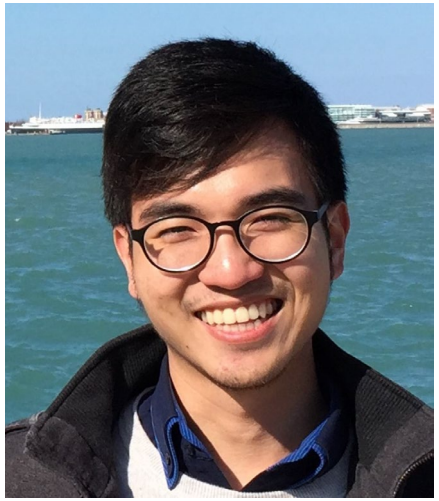
Badih Ghazi

Pritish Kamath



Ravi Kumar

Pasin Manurangsi



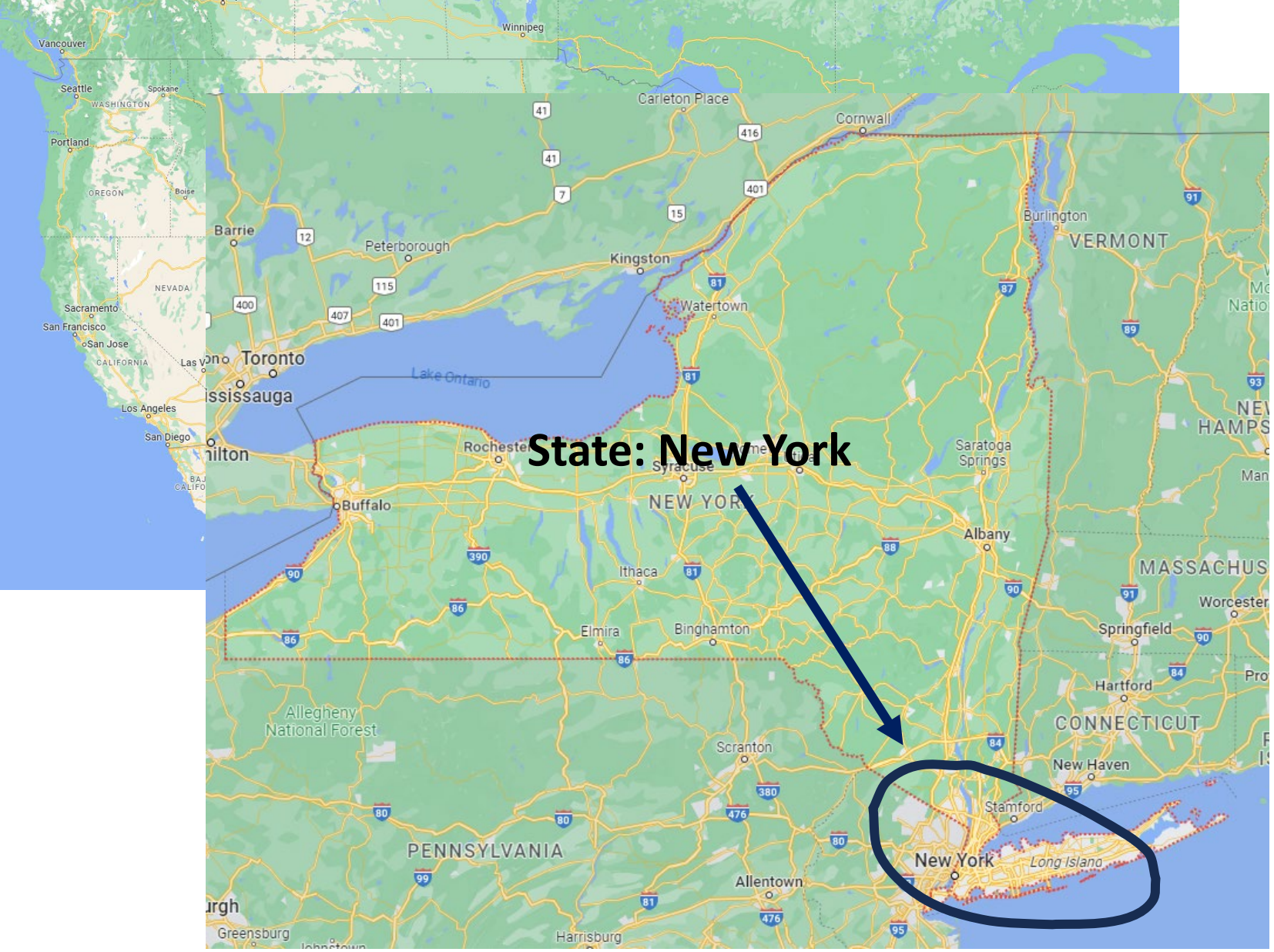
Google Mountain View

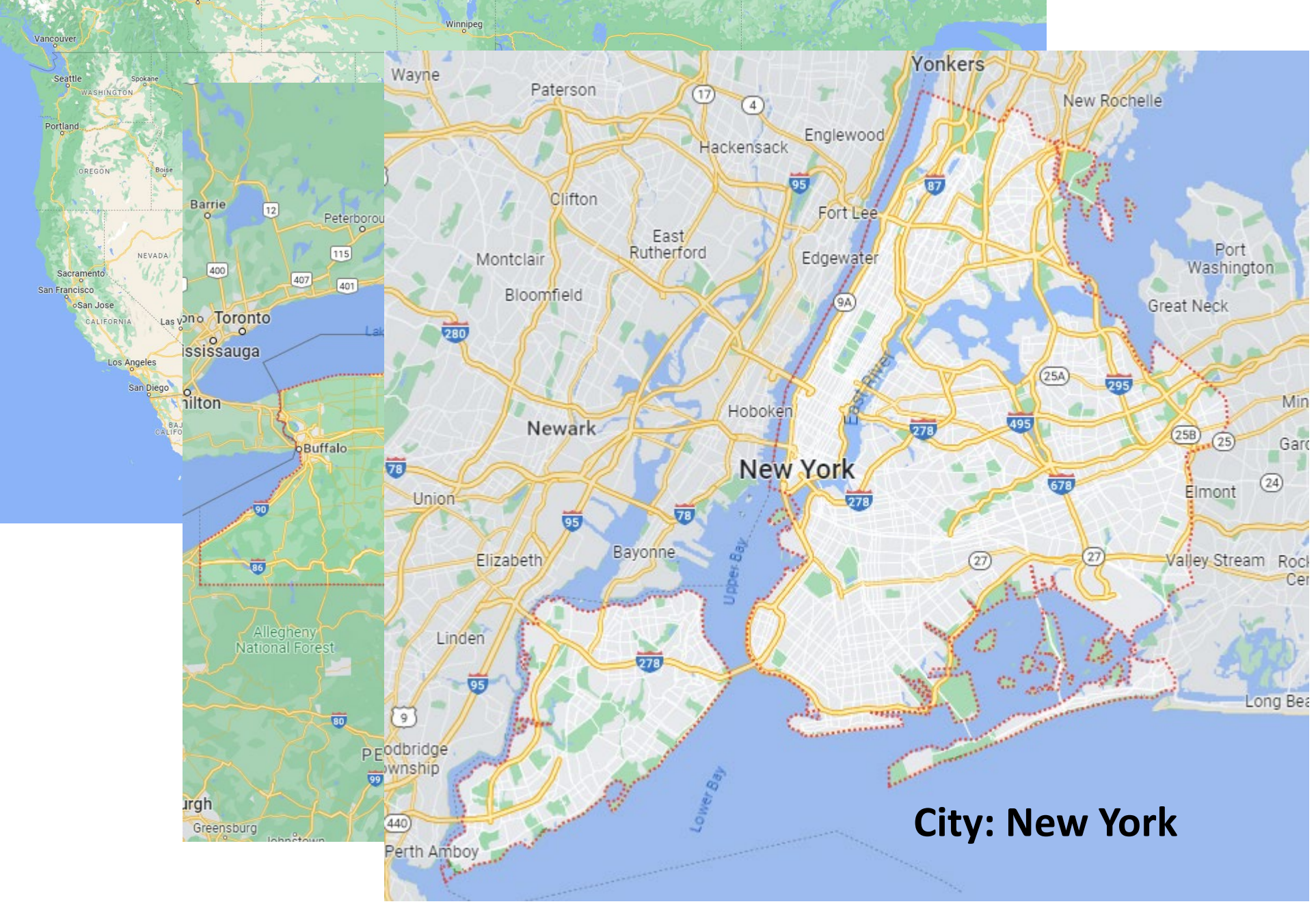
Google Thailand


Country: US
Winnipeg
Vancouver
Seattle
Spokane
WASHINGTON
Portland
MONTANA
Bozeman
Billings
NORTH DAKOTA
MINNESOTA
Minneapolis
SOUTH DAKOTA
WISCONSIN
OREGON
Boise
IDAHO
WYOMING
MICHIGAN
Toronto
Ottawa
Montreal
NEW BRUNSWICK
PRINCE EDWARD ISLAND
MAINE
NOVA SCOTIA
VERMONT
NEW HAMPSHIRE
NEW YORK
MASSACHUSETTS
Chicago
IOWA
NEBRASKA
Omaha
Salt Lake City
ILLINOIS
INDIANA
OHIO
PENNSYLVANIA
New York
Philadelphia
NEVADA
Denver
United States
Kansas City
Indianapolis
Cincinnati
MARYLAND
UTAH
COLORADO
KANSAS
St. Louis
MISSOURI
WEST VIRGINIA
Washington
Sacramento
San Francisco
San Jose
KENTUCKY
VIRGINIA
CALIFORNIA
Las Vegas
Nashville
OKLAHOMA
TENNESSEE
NORTH CAROLINA
Albuquerque
Memphis
Charlotte
ARKANSAS
Los Angeles
ARIZONA
NEW MEXICO
Atlanta
SOUTH CAROLINA
MISSISSIPPI
San Diego
Phoenix
Dallas
ALABAMA
GEORGIA
Bermuda
Tucson
Ciudad Juárez
TEXAS
BAJA CALIFORNIA
SONORA
LOUISIANA
Jacksonville
Houston
San Antonio
New Orleans
CHIHUAHUA
Orlando
Tampa
COAHUILA
FLORIDA
NUEVO LEON
Miami

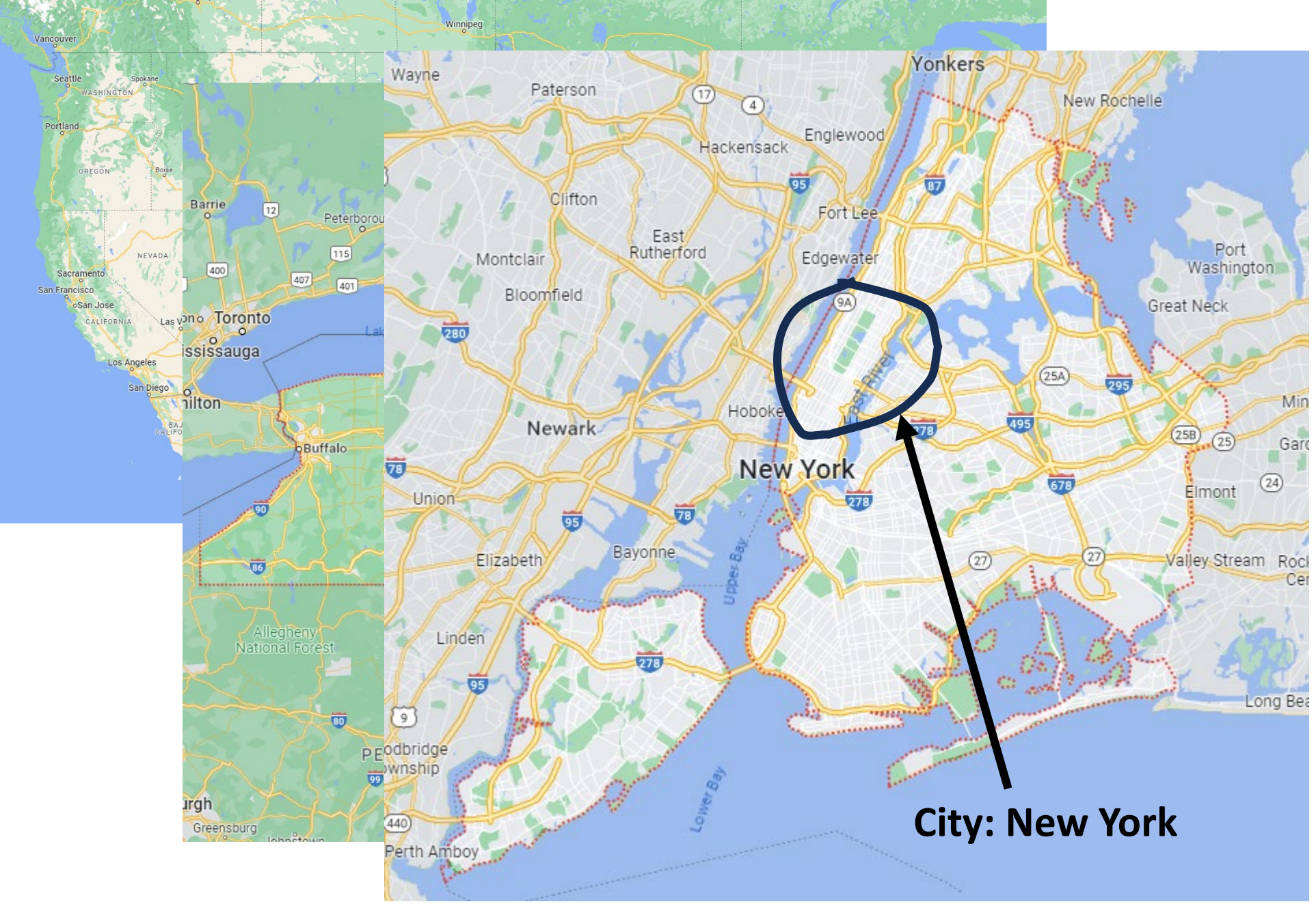

Country: US
Winnipeg
Vancouver
Seattle
Spokane
WASHINGTON
Portland
MONTANA
Bozeman
Billings
NORTH DAKOTA
MINNESOTA
Minneapolis
SOUTH DAKOTA
WISCONSIN
OREGON
Boise
IDAHO
WYOMING
MICHIGAN
Toronto
Ottawa
Montreal
NEW BRUNSWICK
PRINCE EDWARD ISLAND
NOVA SCOTIA
MAINE
VERMONT
NEW HAMPSHIRE
NEW YORK
MASSACHUSETTS
Chicago
IOWA
NEBRASKA
Omaha
Salt Lake City
New York
ILLINOIS
INDIANA
OHIO
NEVADA
Denvero
United States
Kansas City
Indianapolis
Cincinnati
UTAH
COLORADO
KANSAS
MISSOURI
St. Louis
Sacramento
San Francisco
San Jose
WEST VIRGINIA
Washington
KENTUCKY
VIRGINIA
CALIFORNIA
Las Vegas
Nashville
OKLAHOMA
TENNESSEE
Memphis
NORTH CAROLINA
Albuquerque
Charlotte
ARKANSAS
Los Angeles
ARIZONA
NEW MEXICO
Atlanta
SOUTH CAROLINA
MISSISSIPPI
San Diego
Phoenix
Dallas
ALABAMA
GEORGIA
Tucson
Bermuda
Ciudad Juárez
TEXAS
BAJA CALIFORNIA
Jacksonville
SONORA
LOUISIANA
Houston
San Antonio
New Orleans
CHIHUAHUA
Orlando
Tampa
FLORIDA
COAHUILA
NUEVO LEON
Miami

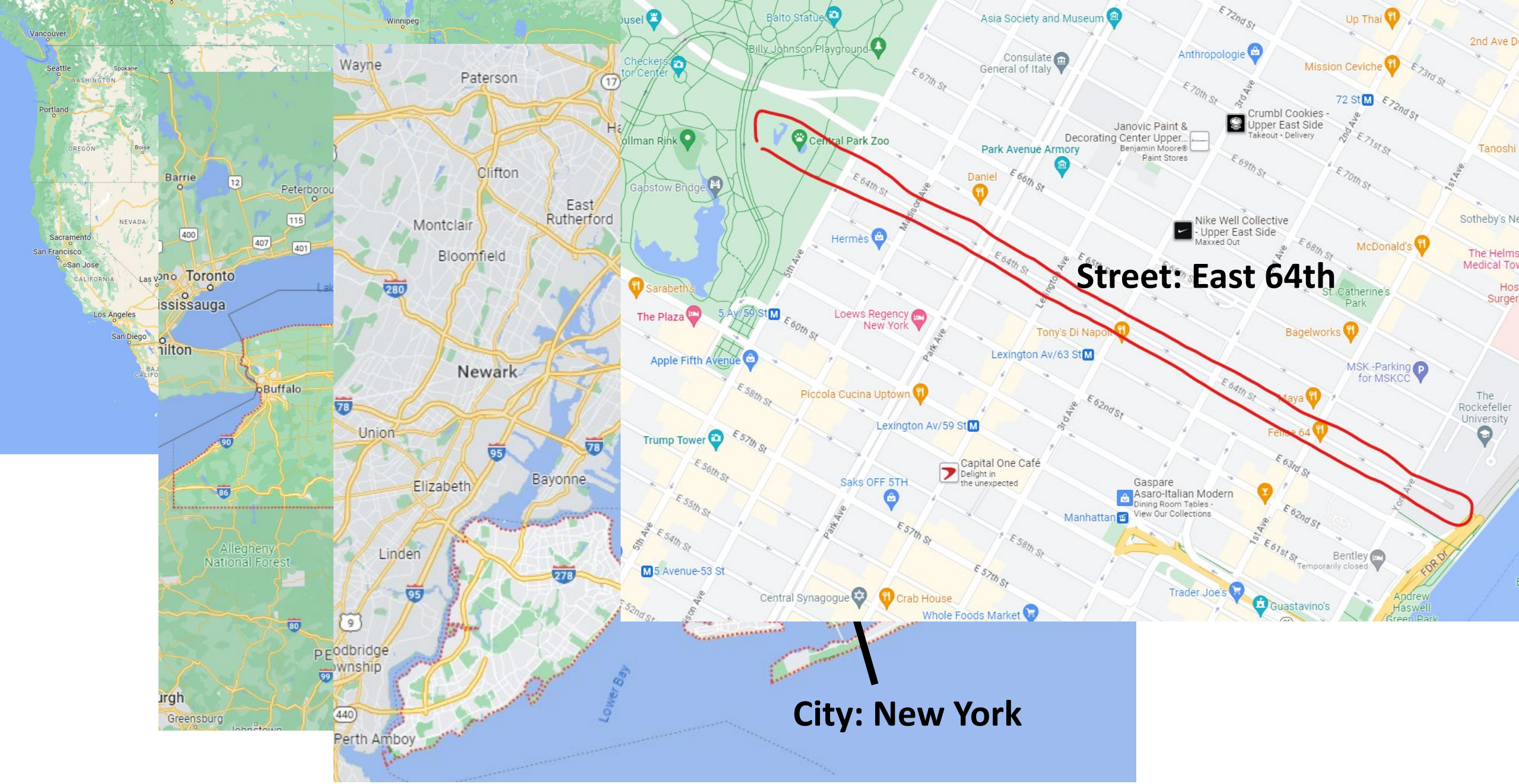

State: New York
Vancouver
Winnipeg
Seattle
Spokane
WASHINGTON
Portland
OREGON
Boise
NEVADA
Sacramento
San Francisco
San Jose
CALIFORNIA
Las V
Los Angeles
San Diego
BAJA CALIFO
Carleton Place
Cornwall
Barrie
Peterborough
Kingston
Watertown
Toronto
Lake Ontario
ississauga
ilton
Rochester
Syracuse
Buffalo
NEW YORK
Ithaca
Elmira
Binghamton
Allegheny National Forest
Scranton
PENNSYLVANIA
Allentown
Harrisburg
Greensburg
urgh
Burlington
VERMONT
Saratoga Springs
Albany
MASSACHUS
Worcester
Springfield
Hartford
CONNECTICUT
New Haven
Stamford
New York
Long Island
NEW HAMPS
Man
Mo
Natio
Pro
41
7
15
416
401
12
115
400
407
81
87
89
91
93
90
88
390
86
476
380
80
78
95
99
84

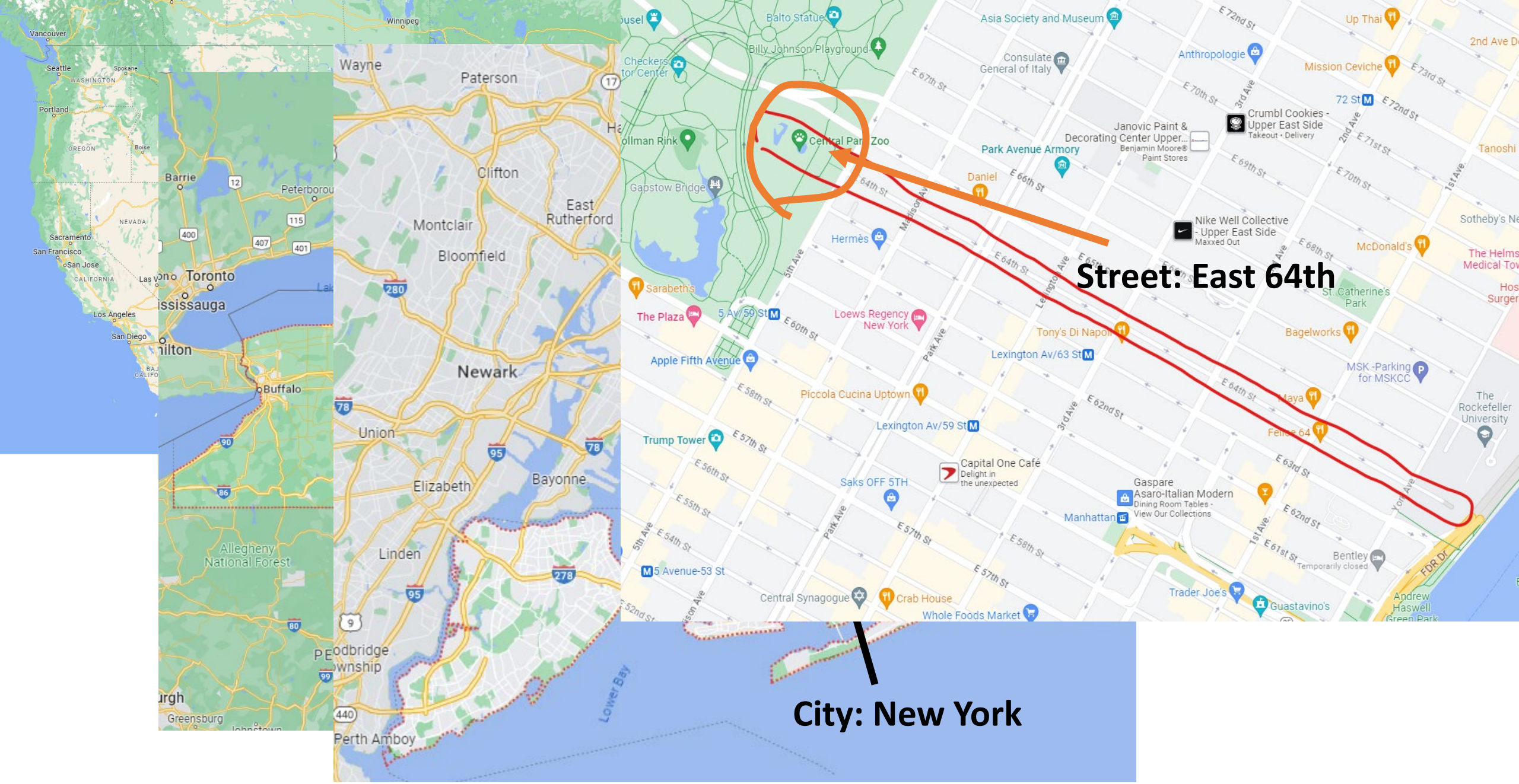

State: New York
Vancouver
Winnipeg
Seattle
Spokane
WASHINGTON
Portland
OREGON
Boise
NEVADA
Sacramento
San Francisco
San Jose
CALIFORNIA
Las V
Los Angeles
San Diego
Carleton Place
Cornwall
Barrie
Peterborough
Kingston
Watertown
Burlington
VERMONT
Toronto
Lake Ontario
ississauga
Rochester
Syracuse
NEW YORK
Buffalo
Saratoga Springs
Albany
Ithaca
Elmira
Binghamton
MASSACHUS
Worcester
Springfield
Hartford
CONNECTICUT
New Haven
Stamford
New York
Long Island
Allegheny National Forest
Scranton
PENNSYLVANIA
Allentown
Harrisburg
Greensburg

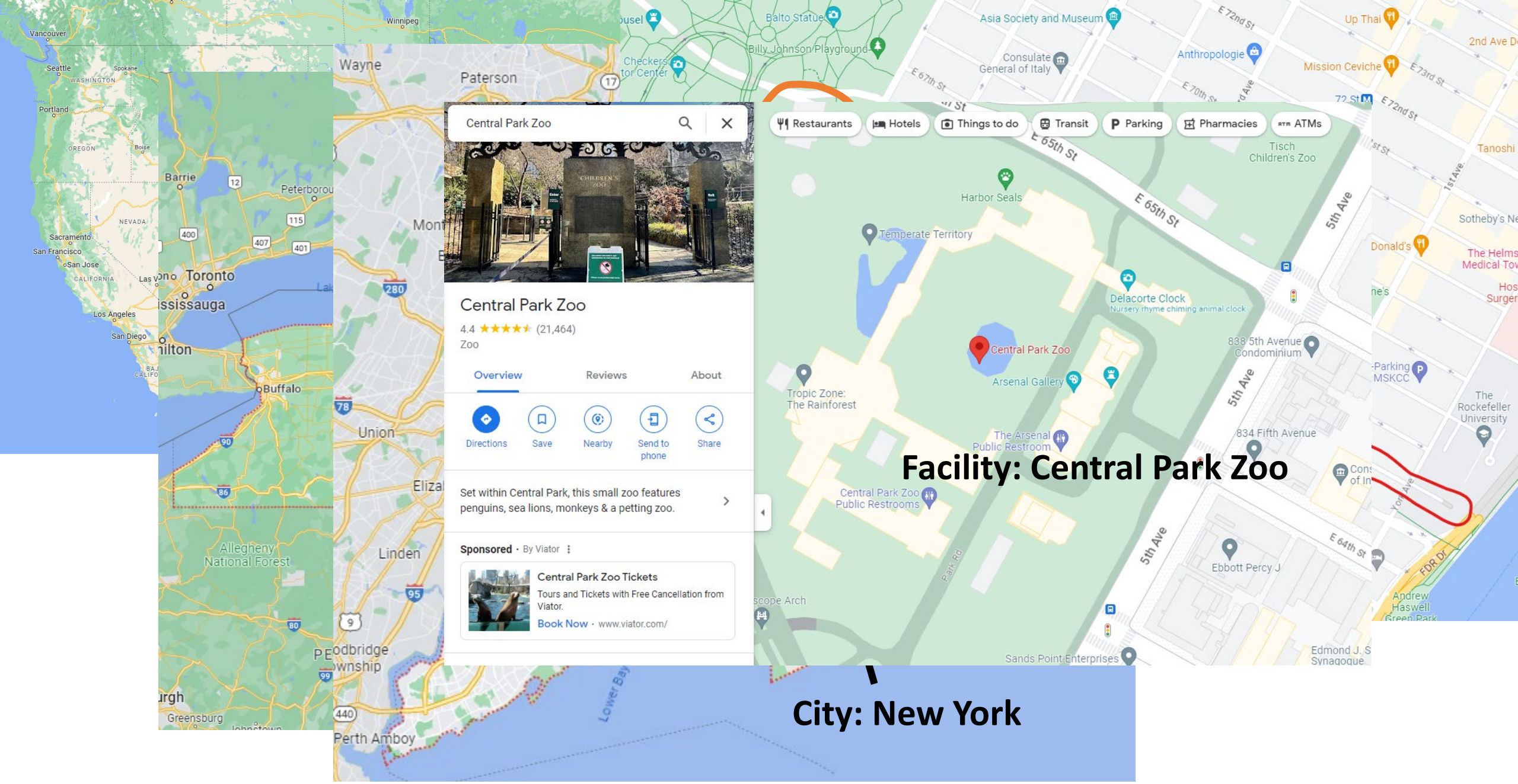


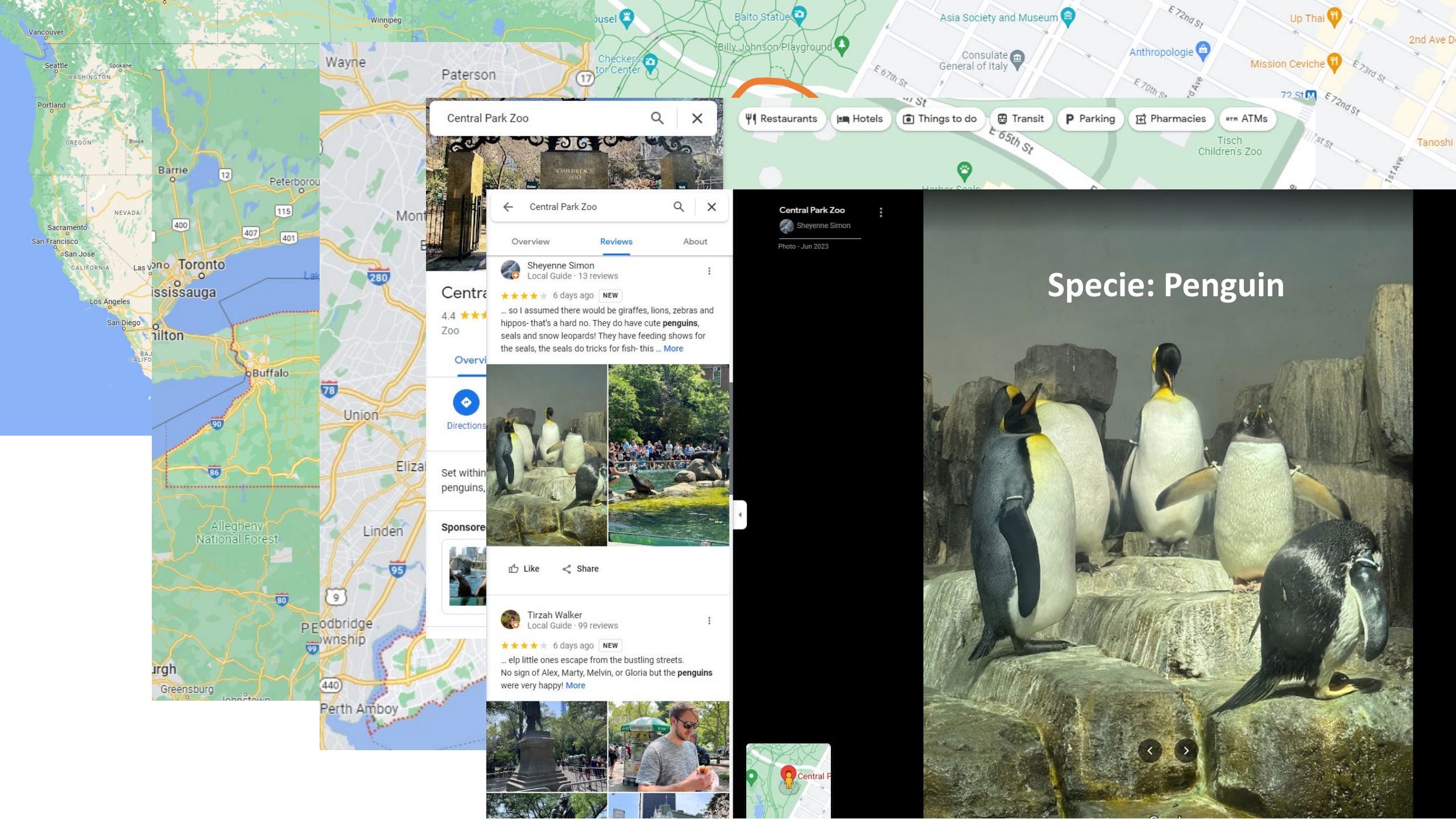
City: New York


Vancouver
Winnipeg
Seattle
Spokane
WASHINGTON
Portland
OREGON
Boise
NEVADA
Sacramento
San Francisco
San Jose
CALIFORNIA
Las V
Los Angeles
San Diego
Barrie
Peterborou
Toronto
ississauga
ilton
Buffalo
Allegheny
National Forest
PE
odbridge
wnship
urgh
Greensburg
Wayne
Paterson
Hackensack
Englewood
Yonkers
New Rochelle
Clifton
Fort Lee
East
Rutherford
Montclair
Edgewater
Port
Washington
Bloomfield
Great Neck
Hoboke
Newark
New York
Union
Elizabeth
Bayonne
Upper Bay
Elmont
Valley Stream
Linden
Lower Bay
Perth Amboy
Long Bea
City: New York

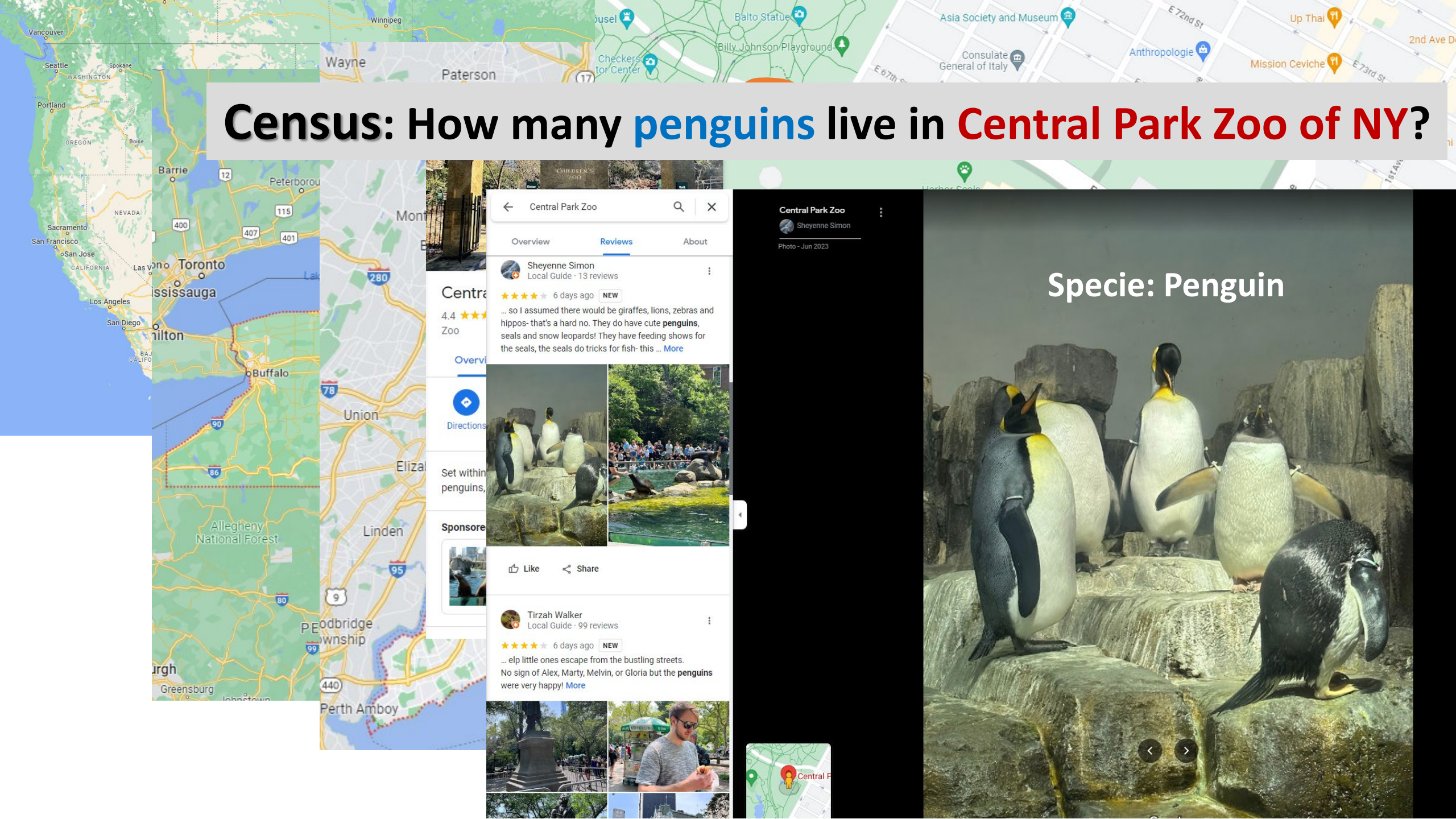

Vancouver
Winnipeg
Seattle
Spokane
WASHINGTON
Portland
OREGON
Boise
NEVADA
Sacramento
San Francisco
San Jose
CALIFORNIA
Las Vegas
Los Angeles
San Diego
Barrie
Peterborough
Toronto
Mississauga
Hamilton
Buffalo
Allegheny National Forest
Greensburg
Wayne
Paterson
Clifton
East Rutherford
Montclair
Bloomfield
Newark
Union
Elizabeth
Bayonne
Linden
Woodbridge Township
Perth Amboy
Lower Bay
Balto Statue
Billy Johnson Playground
Checkers tor Center
Asia Society and Museum
Consulate General of Italy
Anthropologie
Up Thai
Mission Ceviche
72 St
Crumbl Cookies - Upper East Side
Takeout • Delivery
Janovic Paint & Decorating Center Upper...
Benjamin Moore® Paint Stores
Central Park Zoo
Park Avenue Armory
Daniel
Gapstow Bridge
Hermès
Nike Well Collective - Upper East Side
Maxxed Out
McDonald's
Sotheby's
Sarabeth's
The Plaza
5 Av/59 St
Loews Regency New York
St. Catherine's Park
Bagelworks
Tony's Di Napoli
Lexington Av/63 St
Apple Fifth Avenue
MSK -Parking for MSKCC
Piccola Cucina Uptown
Maya
Lexington Av/59 St
Trump Tower
Capital One Café
Delight in the unexpected
Saks OFF 5TH
Gaspare Asaro-Italian Modern
Dining Room Tables - View Our Collections
Manhattan
Bentley
Temporarily closed
5 Avenue-53 St
Central Synagogue
Crab House
Whole Foods Market
Trader Joe's
Guastavino's
Andrew Haswell Green Park
FDR Dr
Tanoshi
The Rockefeller University
Street: East 64th
City: New York


Street: East 64th
City: New York

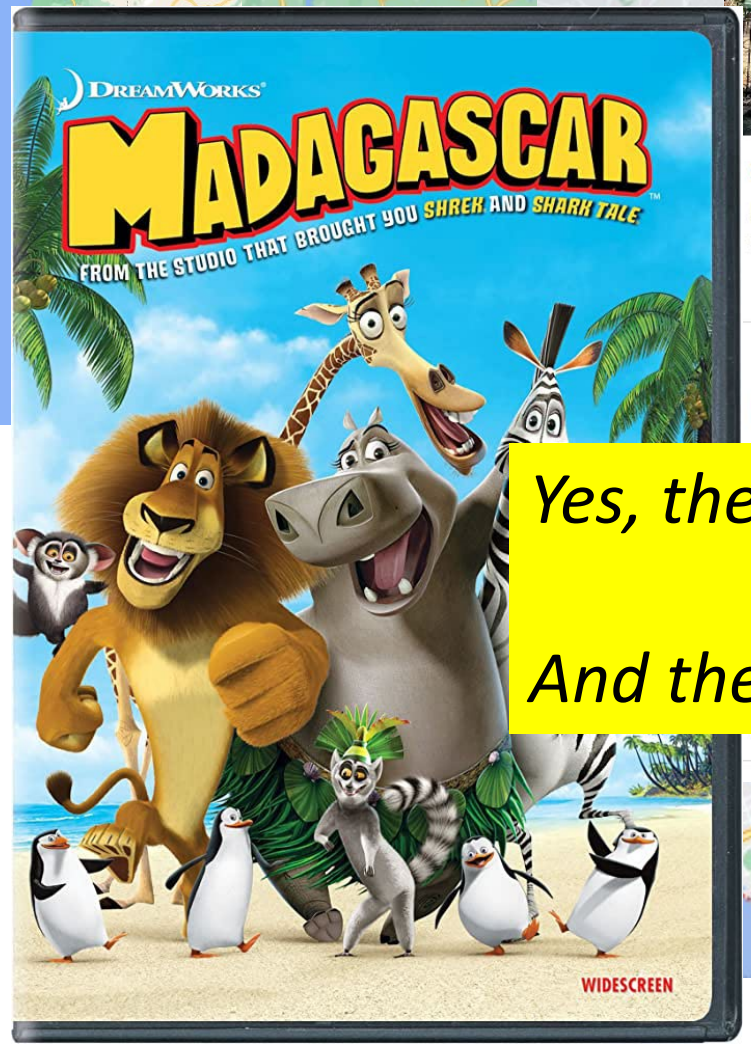

Central Park Zoo
Central Park Zoo
4.4 (21,464)
Zoo
Overview
Reviews
About
Directions
Save
Nearby
Send to phone
Share
Set within Central Park, this small zoo features penguins, sea lions, monkeys & a petting zoo.
Sponsored · By Viator
Central Park Zoo Tickets
Tours and Tickets with Free Cancellation from Viator.
Book Now · www.viator.com/
Restaurants
Hotels
Things to do
Transit
Parking
Pharmacies
ATMs
Harbor Seals
Temperate Territory
Delacorte Clock
Nursery rhyme chiming animal clock
Central Park Zoo
Arsenal Gallery
Tropic Zone: The Rainforest
The Arsenal Public Restroom
Central Park Zoo Public Restrooms
Tisch Children's Zoo
838 5th Avenue Condominium
834 Fifth Avenue
Ebbott Percy J
Sands Point Enterprises
E 65th St
5th Ave
E 64th St
Facility: Central Park Zoo
City: New York

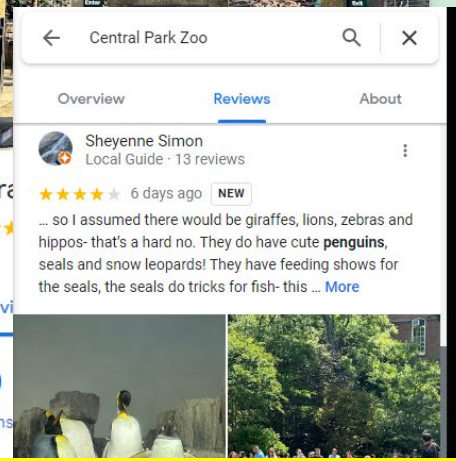

Central Park Zoo
Overview
Reviews
About
Sheyenne Simon
Local Guide · 13 reviews
6 days ago
NEW
... so I assumed there would be giraffes, lions, zebras and hippos- that's a hard no. They do have cute penguins, seals and snow leopards! They have feeding shows for the seals, the seals do tricks for fish- this ... More
Like
Share
Tirzah Walker
Local Guide · 99 reviews
6 days ago
NEW
... elp little ones escape from the bustling streets. No sign of Alex, Marty, Melvin, or Gloria but the penguins were very happy! More
Restaurants
Hotels
Things to do
Transit
Parking
Pharmacies
ATMs
Specie: Penguin

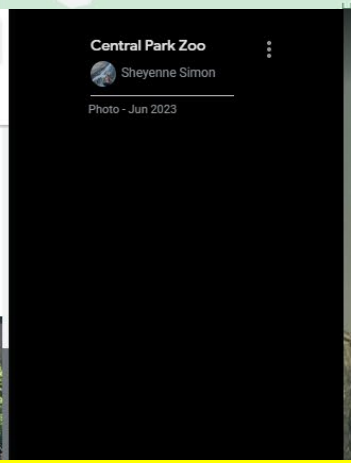
Census: How many penguins live in Central Park Zoo of NY?

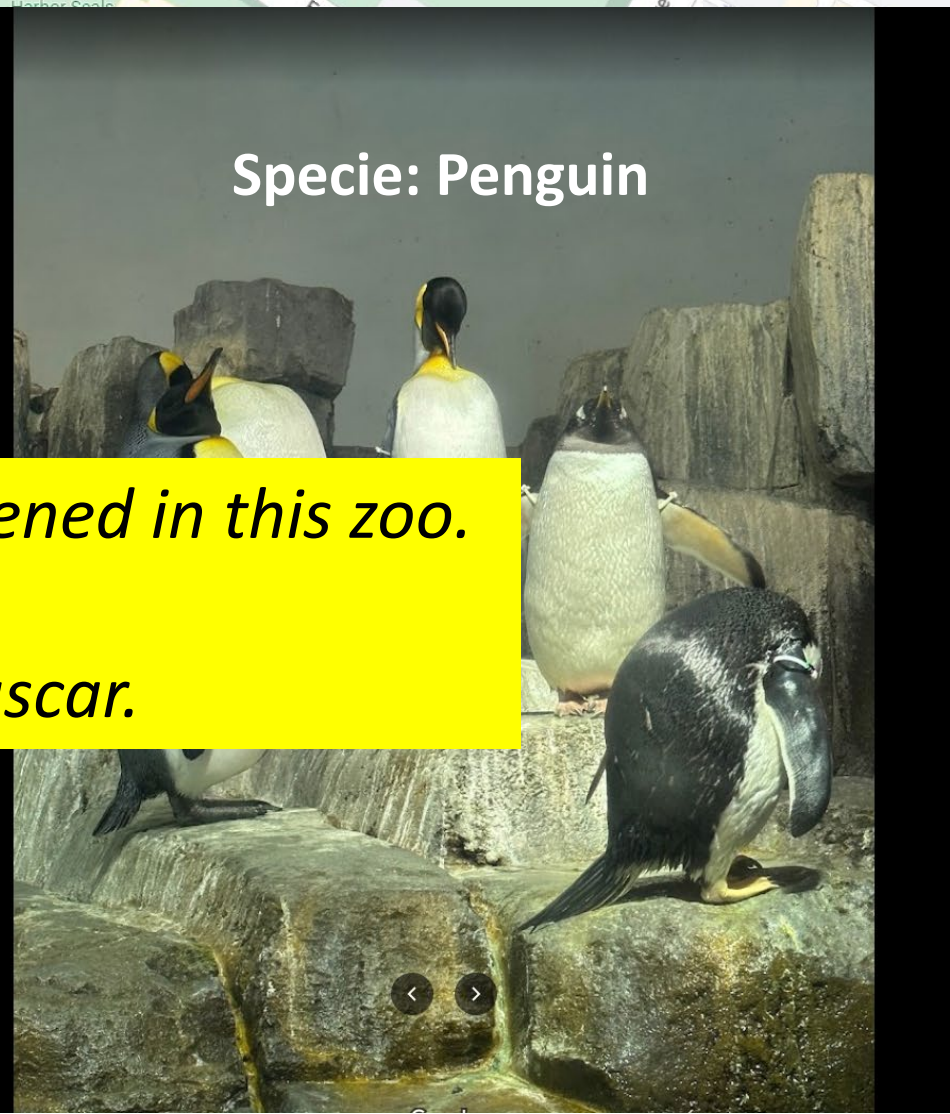


Census: How many penguins live in Central Park Zoo of NY?



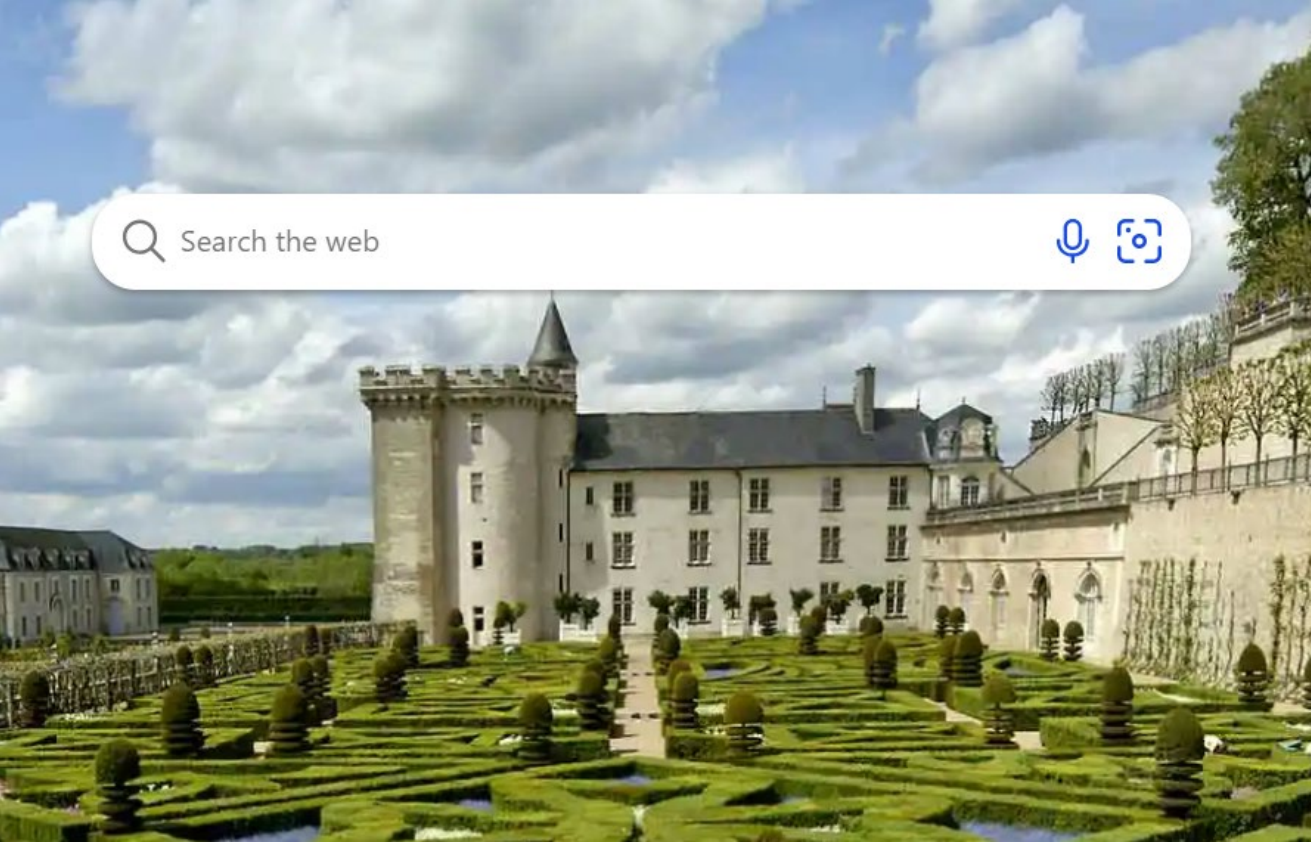






Yes, the story of Madagascar happened in this zoo.

And there is no penguin in Madagascar.

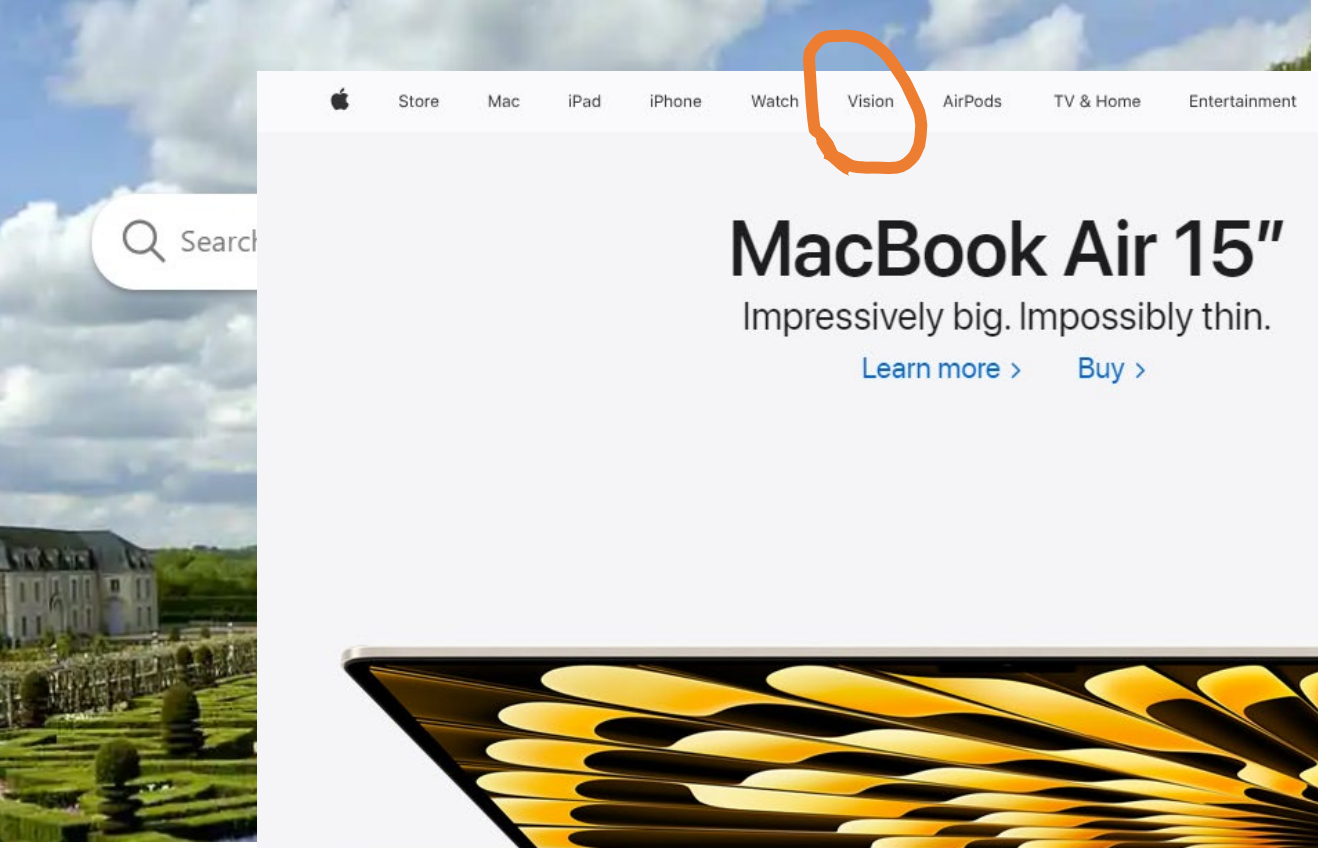

Search the web

Store Mac iPad iPhone Watch Vision AirPods TV & Home Entertainment Accessories Support

MacBook Air 15″

Impressively big. Impossibly thin.

Learn more › Buy ›

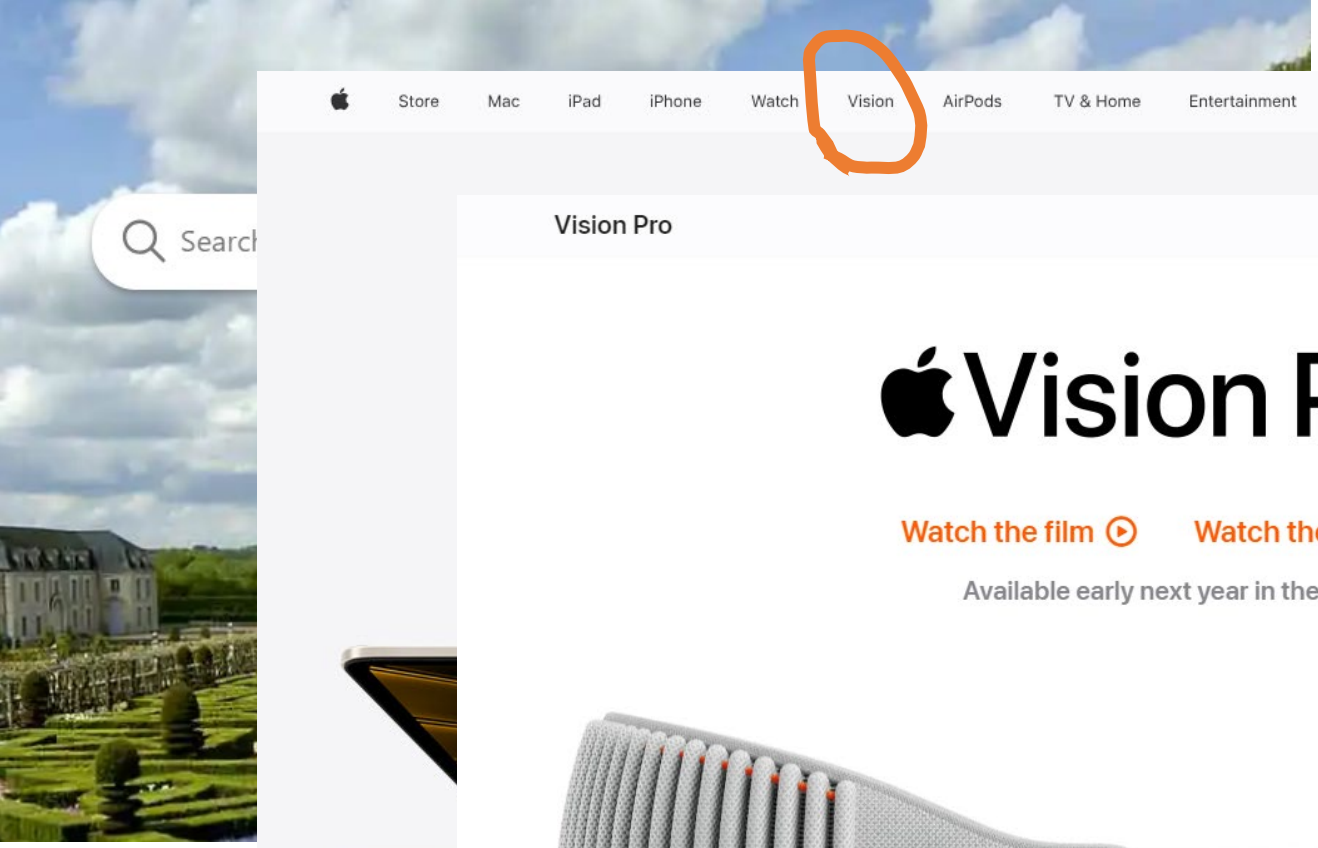


Store Mac iPad iPhone Watch Vision AirPods TV & Home Entertainment Accessories Support

MacBook Air 15″

Impressively big. Impossibly thin.

Learn more › Buy ›



Search

Store Mac iPad iPhone Watch Vision AirPods TV & Home Entertainment Accessories Support

Vision Pro

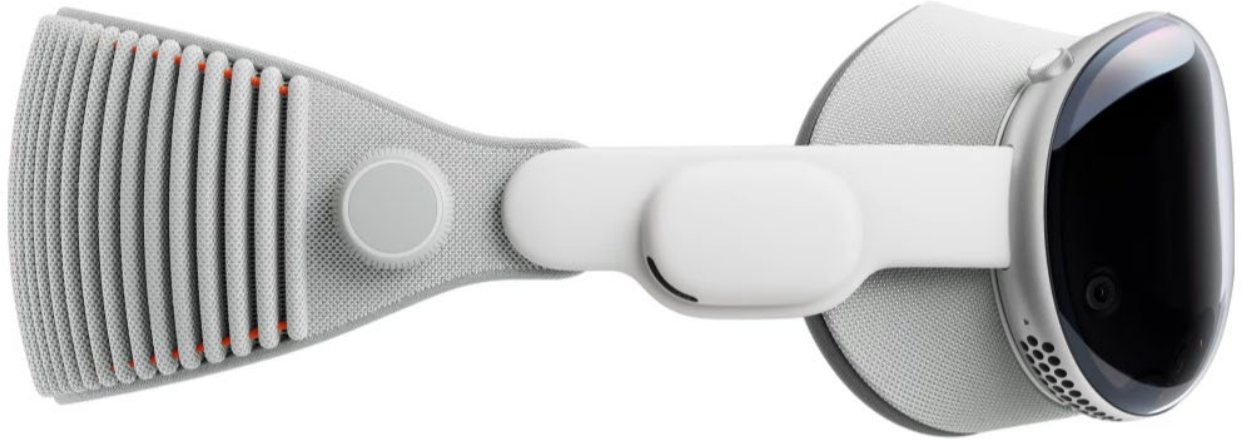
Notify me

Vision Pro

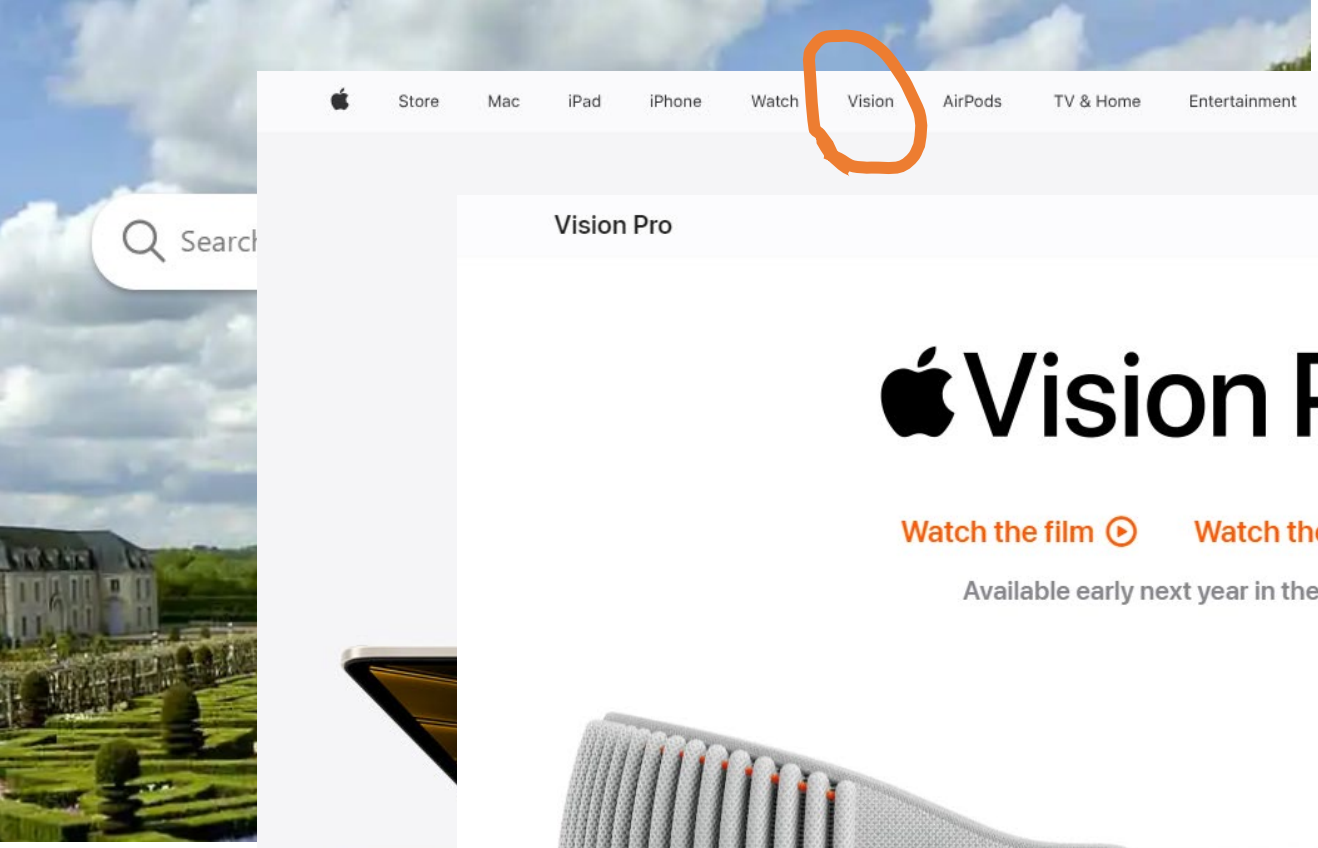
Watch the film

Watch the event >

Available early next year in the U.S.



+ Take a closer look at Vision Pro



Search

Store Mac iPad iPhone Watch Vision AirPods TV & Home Entertainment Accessories Support

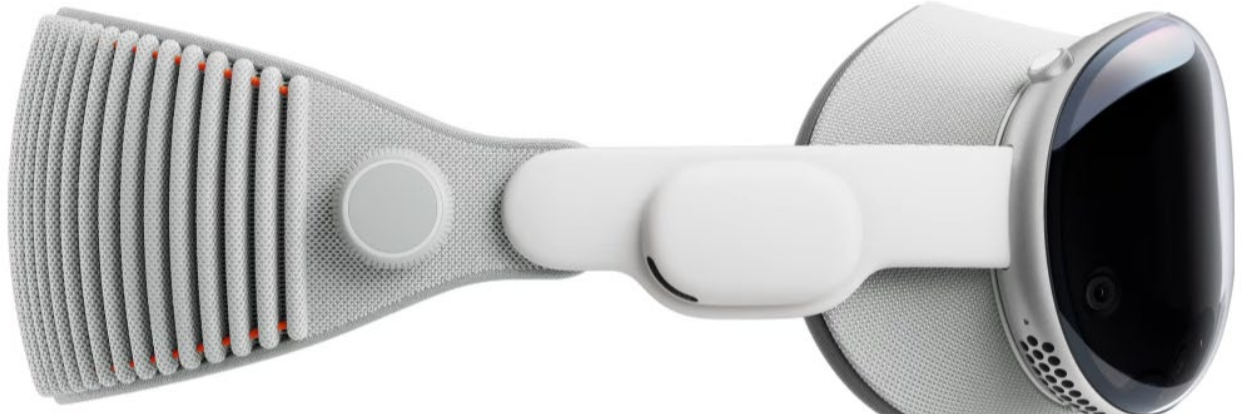
Vision Pro

Notify me

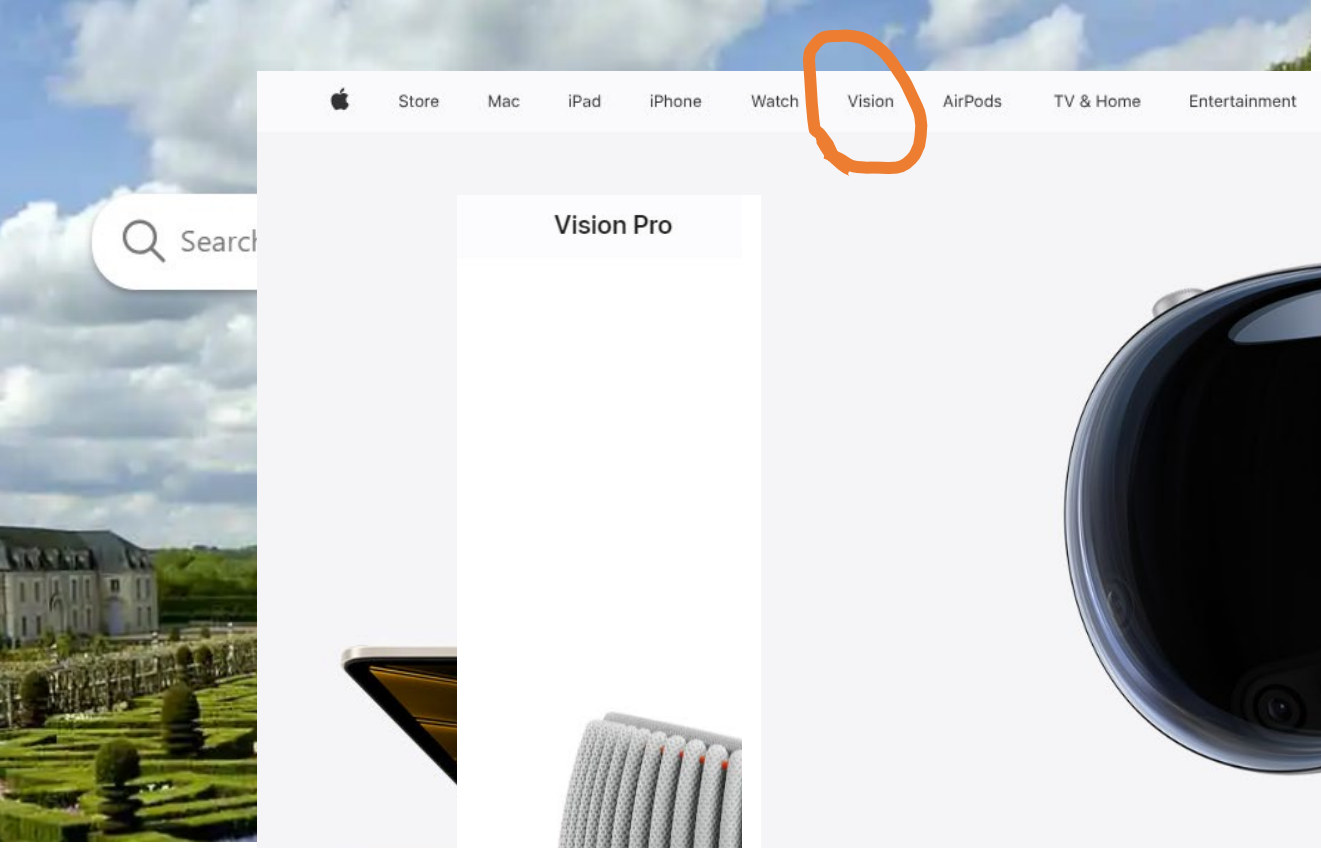
Vision Pro

Watch the film Watch the event ›

Available early next year in the U.S.



+ Take a closer look at Vision Pro



Search

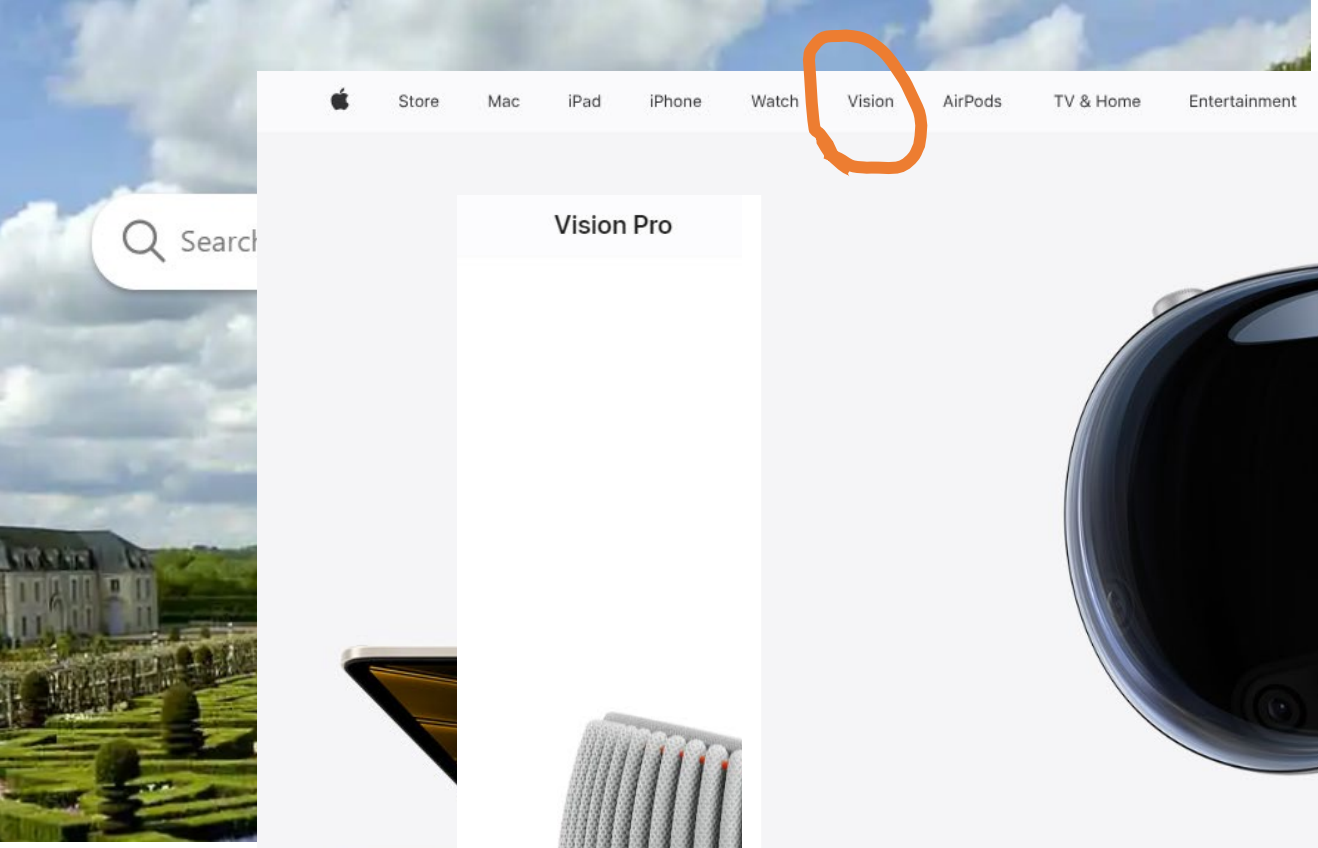
Store Mac iPad iPhone Watch Vision AirPods TV & Home Entertainment Accessories Support

Vision Pro

Enclosure | Cameras and sensors | Audio Straps | Head Band | Displays | Light Seal | Digital Crown | Top button | Power

A singular piece of three-dimensionally formed laminated glass flows into an aluminum alloy frame that gently curves to wrap around your face.

+ Take a closer look at Vision Pro



Search

Store Mac iPad iPhone Watch Vision AirPods TV & Home Entertainment Accessories Support

Vision Pro

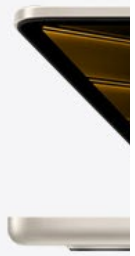
Enclosure | Cameras and sensors | Audio Straps | Head Band | Displays | Light Seal | Digital Crown | Top button | Power

A singular piece of three-dimensionally formed laminated glass flows into an aluminum alloy frame that gently curves to wrap around your face.

+ Take a closer look at Vision Pro

Search

Vision Pro



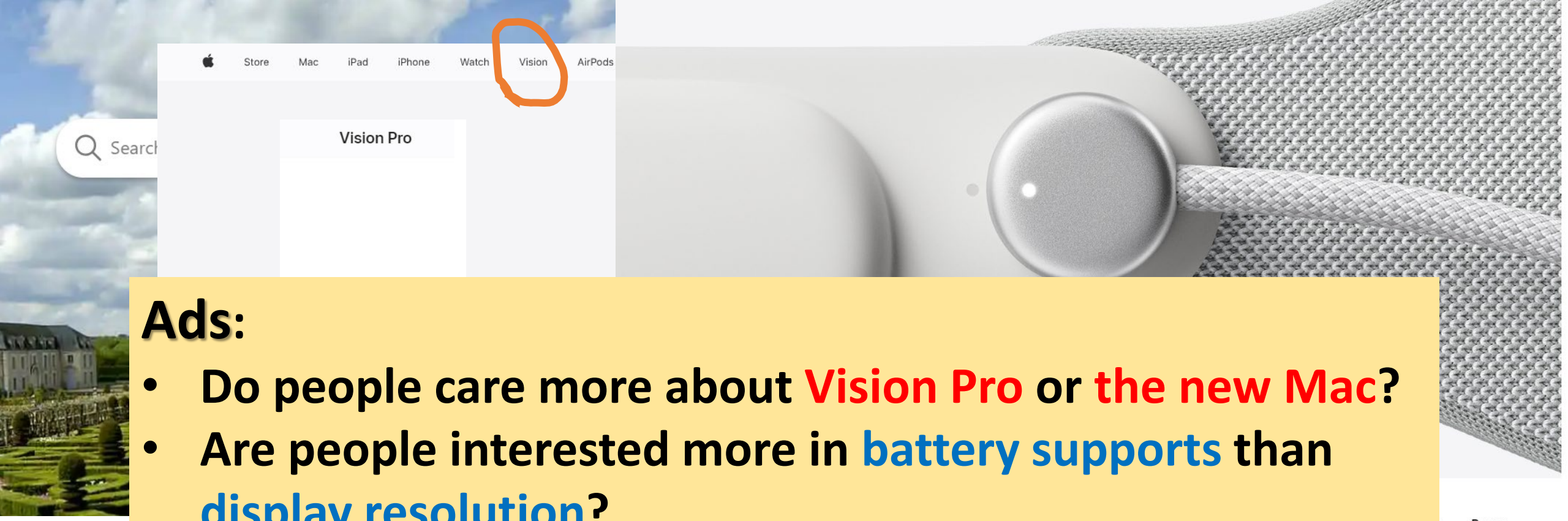
Enclosure | Cameras and sensors | Audio Straps | Head Band | Displays | Light Seal | Digital Crown | Top button | Power

The external battery supports up to 2 hours of use, and all-day use when plugged in.[1]

Enclosure | Cameras and sensors | Audio Straps | Head Band | Displays | Light Seal | Digital Crown | Top button | Power

A singular piece of three-dimensionally formed laminated glass flows into an aluminum alloy frame that gently curves to wrap around your face.

+ Take a closer look at Vision Pro



Ads:

- Do people care more about Vision Pro or the new Mac?
- Are people interested more in battery supports than display resolution?

Power

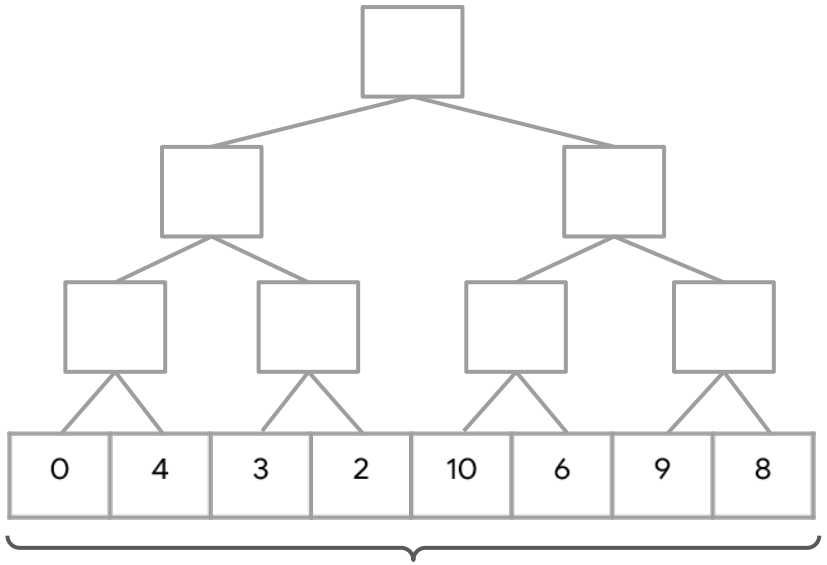
The external battery supports up to 2 hours of use, and all-day use when plugged in.[1]

Enclosure | Cameras and sensors | Audio Straps | Head Band | Displays | Light Seal | Digital Crown | Top button | Power

A singular piece of three-dimensionally formed laminated glass flows into an aluminum alloy frame that gently curves to wrap around your face.

+ Take a closer look at Vision Pro

Counting on Tree

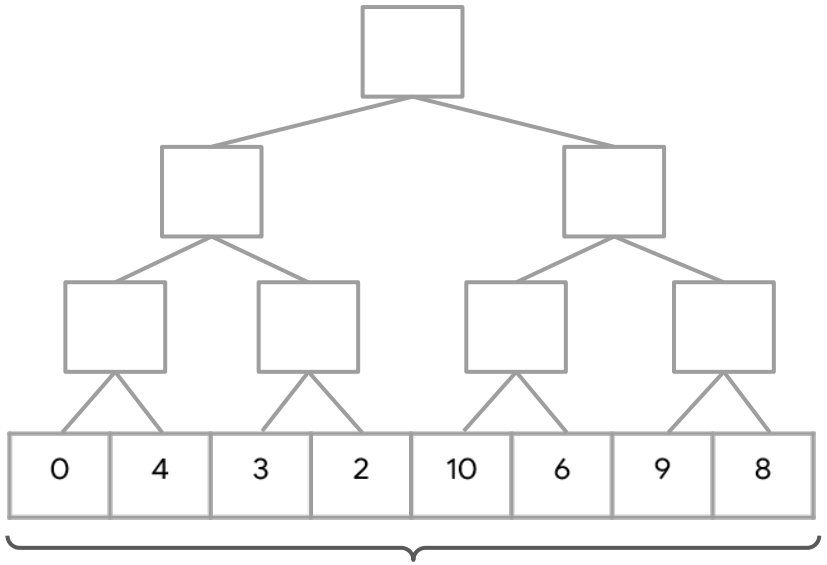


Input: length-N positive vector x

N = # Leaves

d = Tree depth

Counting on Tree



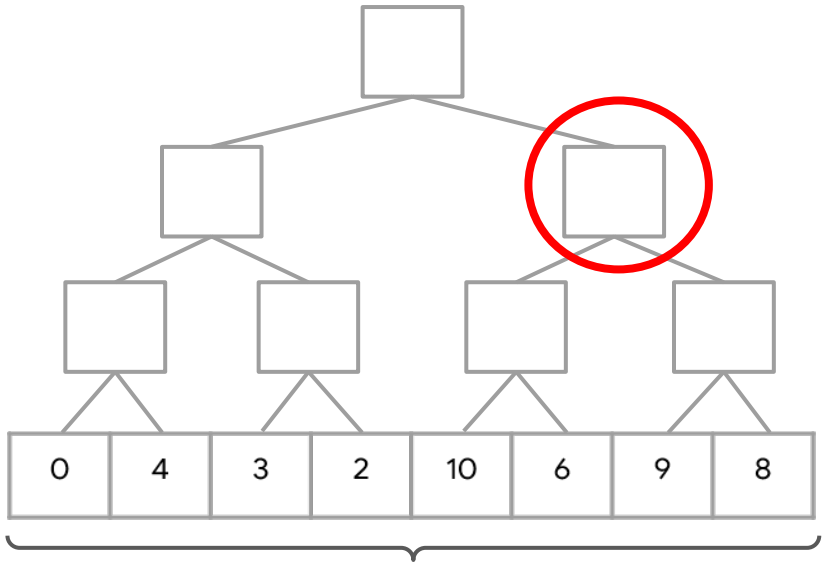
Input: length-N positive vector x

N = # Leaves

d = Tree depth

Node query v:
q_i = total of x_ℓ for all leaves ℓ under v_i

Counting on Tree



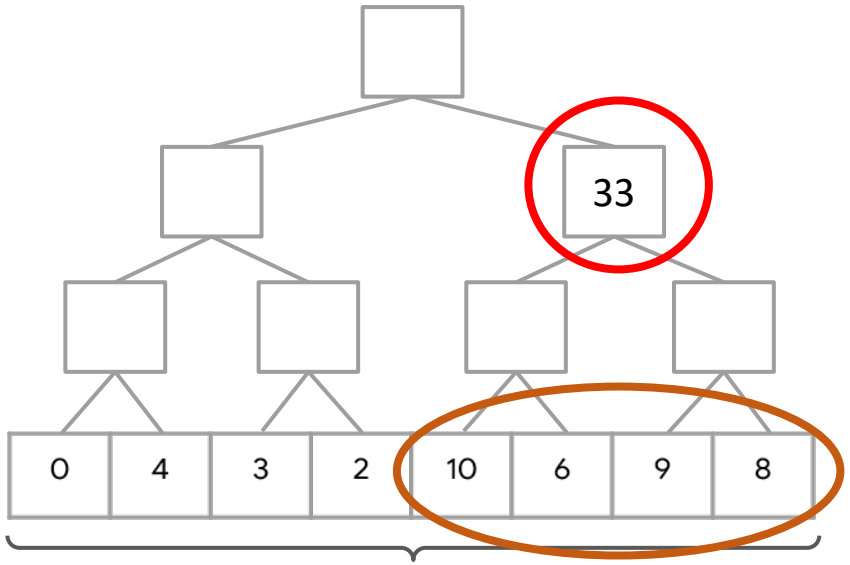
Input: length-N positive vector x

N = # Leaves

d = Tree depth

Node query v:
q_i = total of x_ℓ for all leaves ℓ under v_i

Counting on Tree



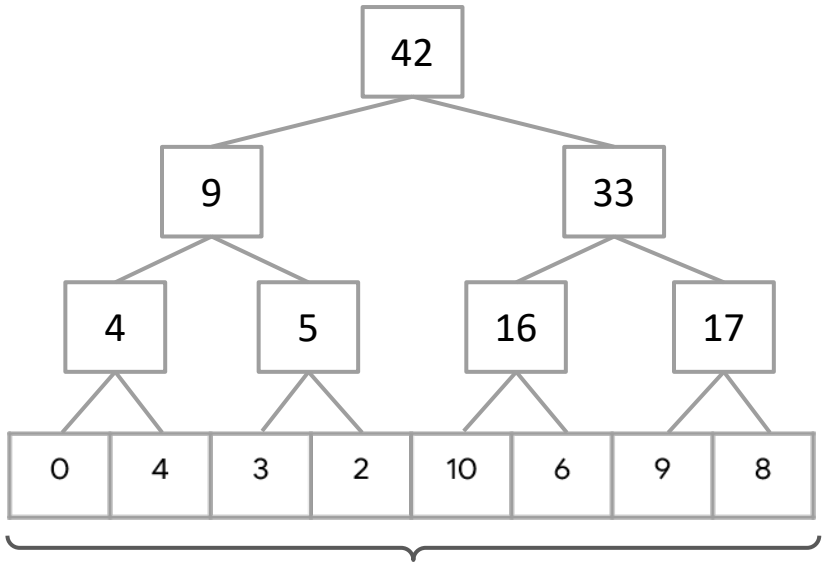
Input: length-N positive vector x

N = # Leaves

d = Tree depth

Node query v:
q_i = total of x_ℓ for all leaves ℓ under v_i

Counting on Tree



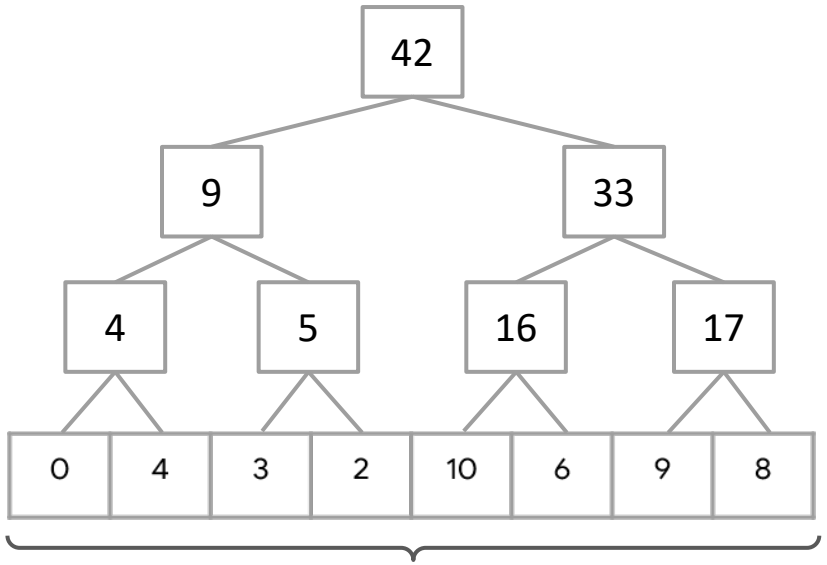
Input: length-N positive vector x

N = # Leaves

d = Tree depth

Node query v:
q_i = total of x_ℓ for all leaves ℓ under v_i

Counting on Tree



Input: length-N positive vector x

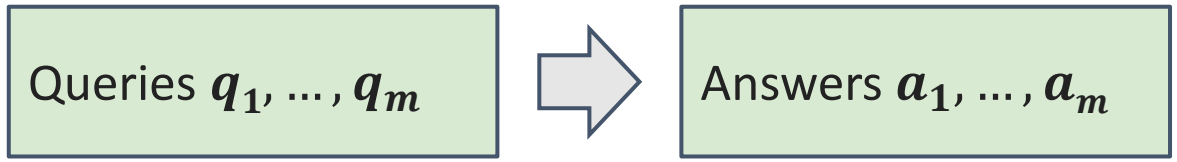
N = # Leaves

d = Tree depth

Node query v:
q_i = total of x_ℓ for all leaves ℓ under v_i

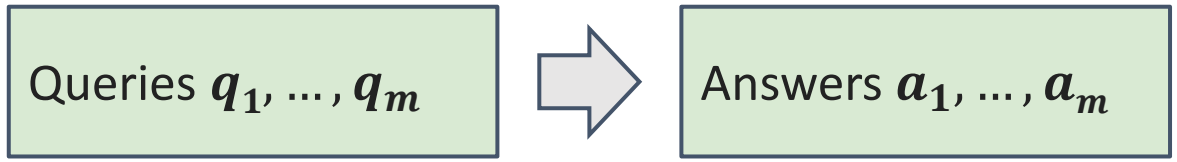
Captures:
- Census (Geographical hierarchy)
- Ads (Webpage hierarchy)
- ……

Accuracy

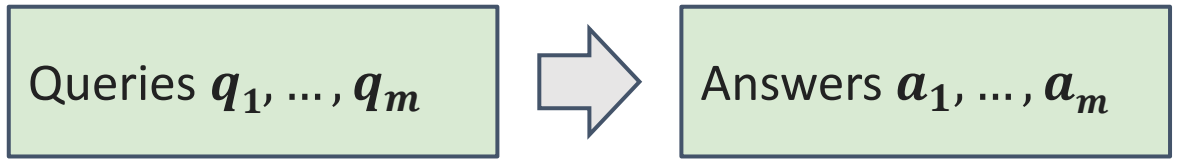


Accuracy

- How many people live in **US**?

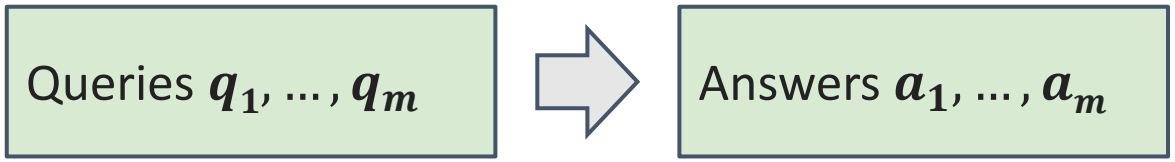


Accuracy



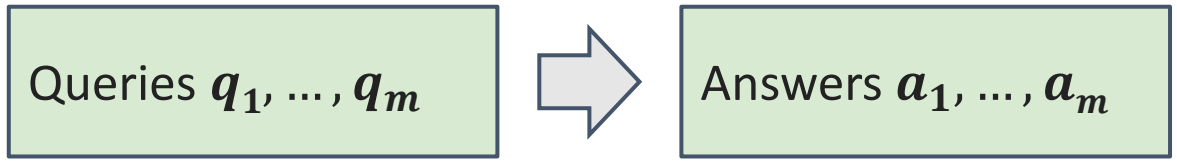
- How many people live in **US**?
- How many people live in **California, US**?
- How many people live in **Berkeley, California, US**?

Accuracy



- How many people live in **US**?
- How many people live in **California, US**?
- How many people live in **Berkeley, California, US**?
- How many people live in **42 Euclidean St, Berkeley, California, US**?
- How many people live in **42 Euclidean St Apt A, Berkeley, California, US**?
- Are they **students**? **Gender**? **Age**?

Accuracy



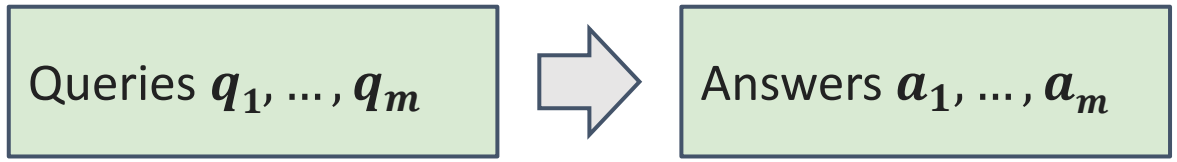
- How many people live in **US**?
- How many people live in **California, US**?
- How many people live in **Berkeley, California, US**?
- How many people live in **42 Euclidean St, Berkeley, California, US**?
- How many people live in **42 Euclidean St Apt A, Berkeley, California, US**?
- Are they **students**? **Gender**? **Age**?

Additive-Only Error

$$\mathrm{err}(a_i) = \sqrt{\mathrm{E}[(a_i - q_i)^2]}$$

$$\mathrm{err} = \max_i \mathrm{err}(e_i)$$

Accuracy



- How many people live in **US**?
- How many people live in **California, US**?
- How many people live in **Berkeley, California, US**?
- How many people live in **42 Euclidean St, Berkeley, California, US**?
- How many people live in **42 Euclidean St Apt A, Berkeley, California, US**?
- Are they **students**? **Gender**? **Age**?

Additive-Only Error

$$\mathrm{err}(\boldsymbol{a_i}) = \sqrt{\mathrm{E}[(\boldsymbol{a_i} - \boldsymbol{q_i})^2]}$$

$$\mathrm{err} = \max_{\boldsymbol{i}} \mathrm{err}(\boldsymbol{e_i})$$

Additive & Multiplicative Error

α : multiplicative factor

$$\mathrm{err}_\alpha(\boldsymbol{a_i}) = \sqrt{\mathrm{E}[\max(|\boldsymbol{a_i} - \boldsymbol{q_i}| - \alpha \boldsymbol{q_i}, 0)^2]}$$

$$\mathrm{err}_\alpha = \max_{i} \mathrm{err}_\alpha(\boldsymbol{a_i})$$

Accuracy

Queries $q_1, \ldots, q_m$ → Answers $a_1, \ldots, a_m$

- How many people live in **US**?
- How many people live in **California, US**?
- How many people live in **Berkeley, C**
- How many people live i **S**?
- How many people li **ia, US**?
- Are they **students**? **G**

Boundary Cases:

- $\alpha = 0$: $\mathrm{err}_\alpha = \mathrm{err}$
- $\alpha \geq 1$: $\mathrm{err}_\alpha = 0$ by $a_i \equiv 0$

Additive-Only Error

$$\mathrm{err}(a_i) = \sqrt{\mathrm{E}[(a_i - q_i)^2]}$$

$$\mathrm{err} = \max_i \mathrm{err}(e_i)$$

Additive & Multiplicative Error

α : multiplicative factor

$$\mathrm{err}_\alpha(a_i) = \sqrt{\mathrm{E}[\max(|a_i - q_i| - \alpha q_i, 0)^2]}$$

$$\mathrm{err}_\alpha = \max_i \mathrm{err}_\alpha(a_i)$$

Differential Privacy (DP) [Dwork-Kenthapadi-McSherry-Mironov-Naor'06]

The output of the algorithm is **probabilistic** and **not sensitive to small perturbations** to the input

Differential Privacy (DP) [Dwork-Kenthapadi-McSherry-Mironov-Naor'06]

The output of the algorithm is **probabilistic** and **not sensitive to small perturbations** to the input

- ⇒ We cannot infer much about the input using the output

Differential Privacy (DP) [Dwork-Kenthapadi-McSherry-Mironov-Naor'06]

The output of the algorithm is **probabilistic** and **not sensitive to small perturbations** to the input

- ⇒ We cannot infer much about the input using the output
- ⇒ Private

Differential Privacy (DP) [Dwork-Kenthapadi-McSherry-Mironov-Naor'06]

The output of the algorithm is **probabilistic** and **not sensitive to small perturbations** to the input

- ⇒ We cannot infer much about the input using the output
- ⇒ Private

(ϵ, δ)-DP: For any neighboring input x, y and any measurable set S

$$\Pr[A(x) \in S] \leq e^{\epsilon} \Pr[A(y) \in S] + \delta$$

Differential Privacy (DP) [Dwork-Kenthapadi-McSherry-Mironov-Naor'06]

The output of the algorithm is **probabilistic** and **not sensitive to small perturbations** to the input

- ⇒ We cannot infer much about the input using the output
- ⇒ Private

(ϵ, δ)-DP: For any neighboring input x, y and any measurable set S

$$\Pr[A(x) \in S] \leq e^{\epsilon} \Pr[A(y) \in S] + \delta$$

x, y are neighbors $\Leftrightarrow |x - y|_1 = 1$

$\delta = 0$: Pure-DP
$\delta \in (0, 1)$: Approximate-DP
$\delta \geq 1$: Trivial

Goal

- Design an algorithm A for the counting task on a depth-$\boldsymbol{d}$ tree

Goal

- Design an algorithm A for the counting task on a depth-$\boldsymbol{d}$ tree
  - Output a vector that is approximately correct under the error metric
    - *Additive & multiplicative error* err_{α}

Goal

- Design an algorithm A for the counting task on a depth-$\boldsymbol{d}$ tree
  - Output a vector that is approximately correct under the error metric
    - *Additive & multiplicative error* err_{α}
- A should be $(\boldsymbol{\epsilon}, \boldsymbol{\delta})$-DP and efficient

Goal

- Design an algorithm A for the counting task on a depth-$\boldsymbol{d}$ tree
  - Output a vector that is approximately correct under the error metric
    - *Additive & multiplicative error* err_α
- A should be $(\boldsymbol{\epsilon}, \boldsymbol{\delta})$-DP and efficient

Smaller err_α	$\Rightarrow$	More accurate
Smaller $\boldsymbol{\epsilon}, \boldsymbol{\delta}$	$\Rightarrow$	More private

Goal

- Design an algorithm A for the counting task on a depth-d tree
  - Output a vector that is approximately correct under the error metric
    - *Additive & multiplicative error* err_α
- A should be (ϵ, δ)-DP and efficient

Smaller err_α	$\Rightarrow$	More accurate
Smaller ϵ, δ	$\Rightarrow$	More private

Fix $\epsilon, \delta, d, \alpha$. What is the smallest possible err_α?

Results

Error / Privacy	Pure-DP $\delta = 0$	Approximate-DP $\delta \in (0, 1)$
Additive only $\alpha = 0$	$O(d/\epsilon)$ [Laplace Mechanism]	$O\left(\sqrt{d \log(1/\delta)}/\epsilon\right)$ [Gaussian Mechanism]
	$\Omega(d/\epsilon)$ [Our result]	$\Omega_{\epsilon,\delta}\left(\sqrt{d}\right)$ [Edmonds-Nikolov-Ullman'20]
Multiplicative & additive $\alpha \in (0, 1)$ is a constant	$\Omega(d/\epsilon)$ [Our result]	$O(\log(d/\delta)/\epsilon)$ [Our result]

Results

Error / Privacy	Pure-DP $\delta = 0$	Approximate-DP $\delta \in (0, 1)$
Additive only $\alpha = 0$	$O(d/\epsilon)$ [Laplace Mechanism]	$O\left(\sqrt{d \log(1/\delta)}/\epsilon\right)$ [Gaussian Mechanism]
	$\Omega(d/\epsilon)$ [Our result]	$\Omega_{\epsilon,\delta}\left(\sqrt{d}\right)$ [Edmonds-Nikolov-Ullman'20]
Multiplicative & additive $\alpha \in (0, 1)$ is a constant	$\Omega(d/\epsilon)$ [Our result]	$O(\log(d/\delta)/\epsilon)$ [Our result]

Remark: Lower bounds are proved for binary trees. Upper bounds hold for arbitrary trees.

Results

Error / Privacy	Pure-DP $\delta = 0$	Approximate-DP $\delta \in (0, 1)$
Additive only $\alpha = 0$	$O(d/\epsilon)$ [Laplace Mechanism]	$O\left(\sqrt{d \log(1/\delta)}/\epsilon\right)$ [Gaussian Mechanism]
	$\Omega(d/\epsilon)$ [Our result]	$\Omega_{\epsilon,\delta}\left(\sqrt{d}\right)$ [Edmonds-Nikolov-Ullman'20]
Multiplicative & additive $\alpha \in (0, 1)$ is a constant	$\Omega(d/\epsilon)$ [Our result]	$O(\log(d/\delta)/\epsilon)$ [Our result]

Simple mechanisms work best

Remark: Lower bounds are proved for binary trees. Upper bounds hold for arbitrary trees.

Results

Error / Privacy	Pure-DP $\delta = 0$	Approximate-DP $\delta \in (0, 1)$
Additive only $\alpha = 0$	$O(d/\epsilon)$ [Laplace Mechanism]	$O\left(\sqrt{d \log(1/\delta)}/\epsilon\right)$ [Gaussian Mechanism]
	$\Omega(d/\epsilon)$ [Our result]	$\Omega_{\epsilon,\delta}\left(\sqrt{d}\right)$ [Edmonds-Nikolov-Ullman'20]
Multiplicative & additive $\alpha \in (0, 1)$ is a constant	$\Omega(d/\epsilon)$ [Our result]	$O(\log(d/\delta)/\epsilon)$ [Our result]

Tight even when we allow multiplicative error

Remark: Lower bounds are proved for binary trees. Upper bounds hold for arbitrary trees.

Results

Error / Privacy	Pure-DP $\delta = 0$	Approximate-DP $\delta \in (0, 1)$
Additive only $\alpha = 0$	$O(d/\epsilon)$ [Laplace Mechanism]	$O\left(\sqrt{d \log(1/\delta)}/\epsilon\right)$ [Gaussian Mechanism]
	$\Omega(d/\epsilon)$ [Our result]	$\Omega_{\epsilon,\delta}($ [Edmonds-Nikol
Multiplicative & additive $\alpha \in (0, 1)$ is a constant	$\Omega(d/\epsilon)$ [Our result]	$O(\log(d/\delta)/\epsilon)$ [Our result]

Tight even when we allow multiplicative error

Exponential improvement when we allow multiplicative error

Remark: Lower bounds are proved for binary trees. Upper bounds hold for arbitrary trees.

Results

Error / Privacy	Pure-DP $\delta = 0$	Approximate-DP $\delta \in (0, 1)$
Additive only $\alpha = 0$	$O(d/\epsilon)$ [Laplace Mechanism]	$O\left(\sqrt{d \log(1/\delta)}/\epsilon\right)$ [Gaussian Mechanism]
	$\Omega(d/\epsilon)$ [Our result]	$\Omega_{\epsilon,\delta}\left(\sqrt{d}\right)$ [Edmonds-Nikolov-Ullman'20]
Multiplicative & additive $\alpha \in (0, 1)$ is a constant	$\Omega(d/\epsilon)$ [Our result]	$O(\log(d/\delta)/\epsilon)$ [Our result]

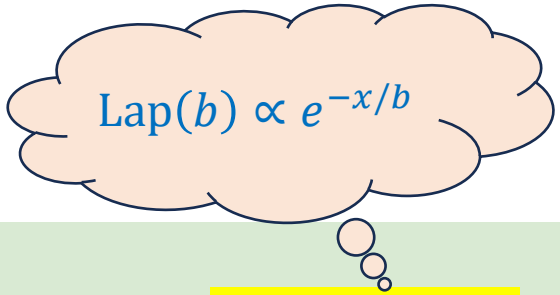
Remark: Lower bounds are proved for binary trees. Upper bounds hold for arbitrary trees.

Laplace / Gaussian Mechanism

[Dwork-Kenthapadi-McSherry-Mironov-Naor’06, Dwork-Roth’16]

Laplace / Gaussian Mechanism

[Dwork-Kenthapadi-McSherry-Mironov-Naor'06, Dwork-Roth'16]



Adding independent **Laplace noise** $\mathbf{Lap}(\boldsymbol{d}/\boldsymbol{\epsilon})$ to each query satisfies $\boldsymbol{\varepsilon}$-DP

Laplace / Gaussian Mechanism

[Dwork-Kenthapadi-McSherry-Mironov-Naor'06, Dwork-Roth'16]

$\mathrm{Lap}(b) \propto e^{-x/b}$

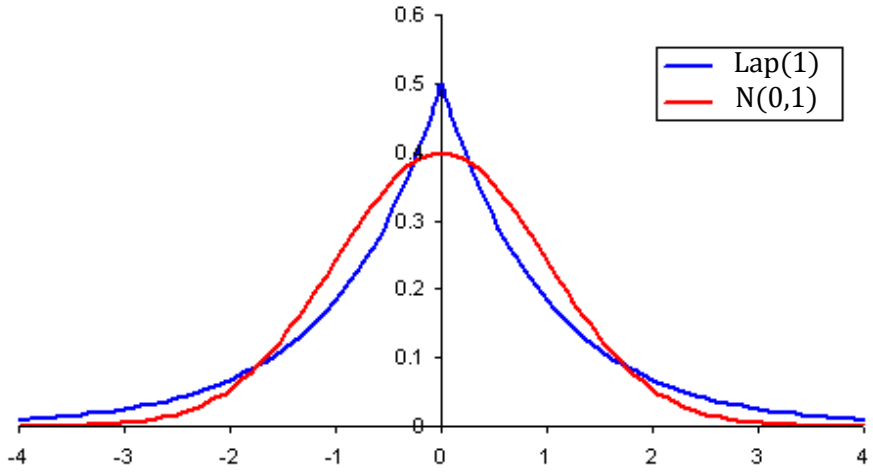
Adding independent **Laplace noise** $\mathbf{Lap}(\boldsymbol{d}/\boldsymbol{\epsilon})$ to each query satisfies $\boldsymbol{\varepsilon}$-DP

Adding independent **Gaussian noise** $\mathbf{N}\left(\mathbf{0}, \boldsymbol{d}\,\mathbf{log}(\mathbf{1}/\boldsymbol{\delta})\,/\boldsymbol{\epsilon}^{\mathbf{2}}\right)$ to each query satisfies $(\boldsymbol{\epsilon}, \boldsymbol{\delta})$-DP

$\mathrm{N}(0, \sigma^2) \propto e^{-x^2/2\sigma^2}$

Laplace / Gaussian Mechanism

[Dwork-Kenthapadi-McSherry-Mironov-Naor'06, Dwork-Roth'16]



$\mathrm{Lap}(b) \propto e^{-x/b}$

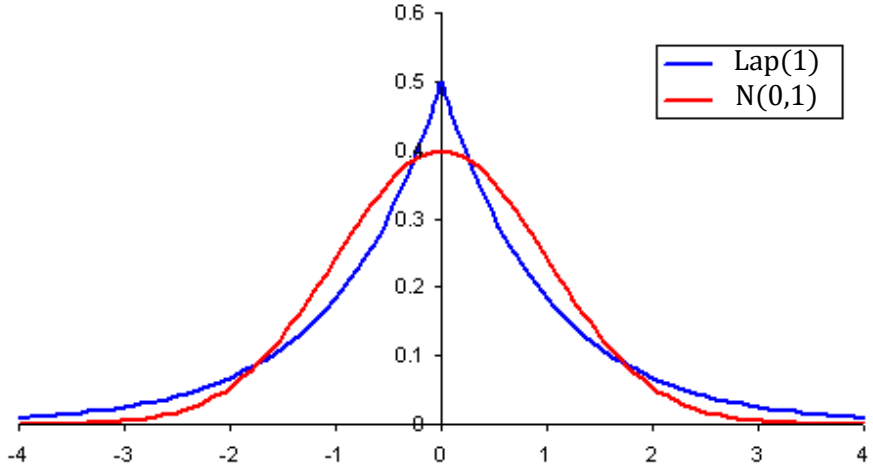
Adding independent **Laplace noise** $\mathbf{Lap}(\boldsymbol{d}/\boldsymbol{\epsilon})$ to each query satisfies $\boldsymbol{\varepsilon}$-DP

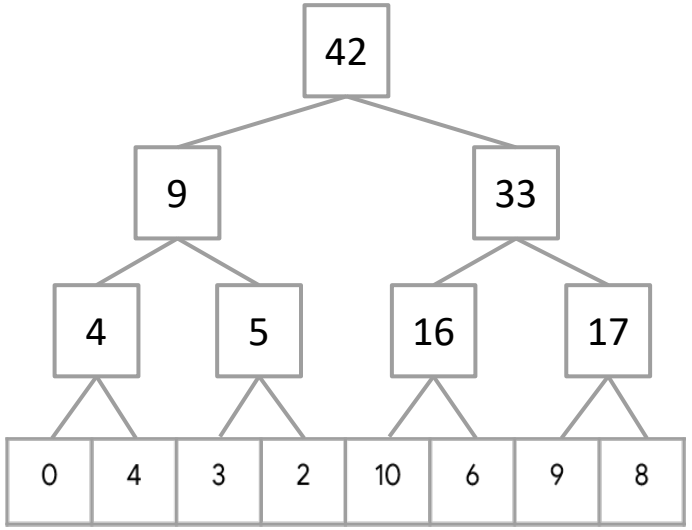
Adding independent **Gaussian noise** $\mathbf{N}\left(\mathbf{0}, \boldsymbol{d}\,\mathbf{log}(\mathbf{1}/\boldsymbol{\delta})\,/\boldsymbol{\epsilon}^{\mathbf{2}}\right)$ to each query satisfies $(\boldsymbol{\epsilon}, \boldsymbol{\delta})$-DP

$\mathrm{N}(0, \sigma^2) \propto e^{-x^2/2\sigma^2}$

Laplace / Gaussian Mechanism

[Dwork-Kenthapadi-McSherry-Mironov-Naor'06, Dwork-Roth'16]





$\text{Lap}(b) \propto e^{-x/b}$

Adding independent **Laplace noise** $\mathbf{Lap}(\boldsymbol{d}/\boldsymbol{\epsilon})$ to each query satisfies $\boldsymbol{\varepsilon}$-DP

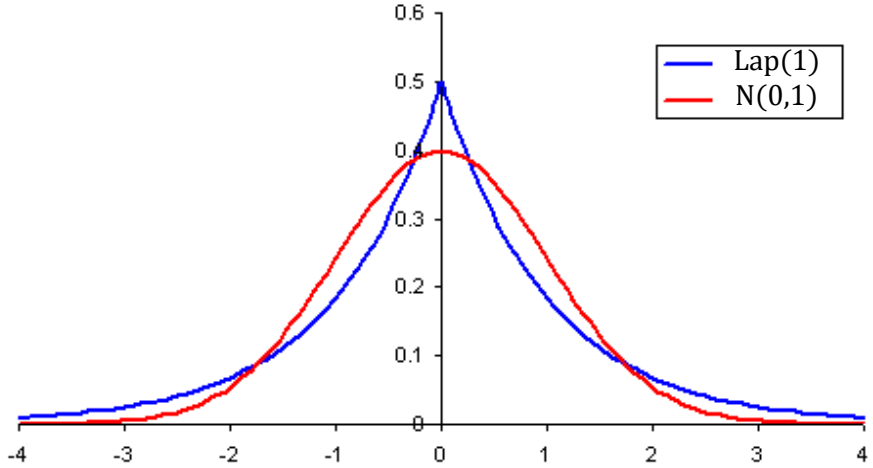
Adding independent **Gaussian noise** $\mathbf{N}\left(\mathbf{0}, \boldsymbol{d}\,\mathbf{log}(\mathbf{1}/\boldsymbol{\delta})\,/\boldsymbol{\epsilon}^{\mathbf{2}}\right)$ to each query satisfies $(\boldsymbol{\epsilon}, \boldsymbol{\delta})$-DP

$\text{N}(0, \sigma^2) \propto e^{-x^2/2\sigma^2}$

Laplace / Gaussian Mechanism

[Dwork-Kenthapadi-McSherry-Mironov-Naor'06, Dwork-Roth'16]

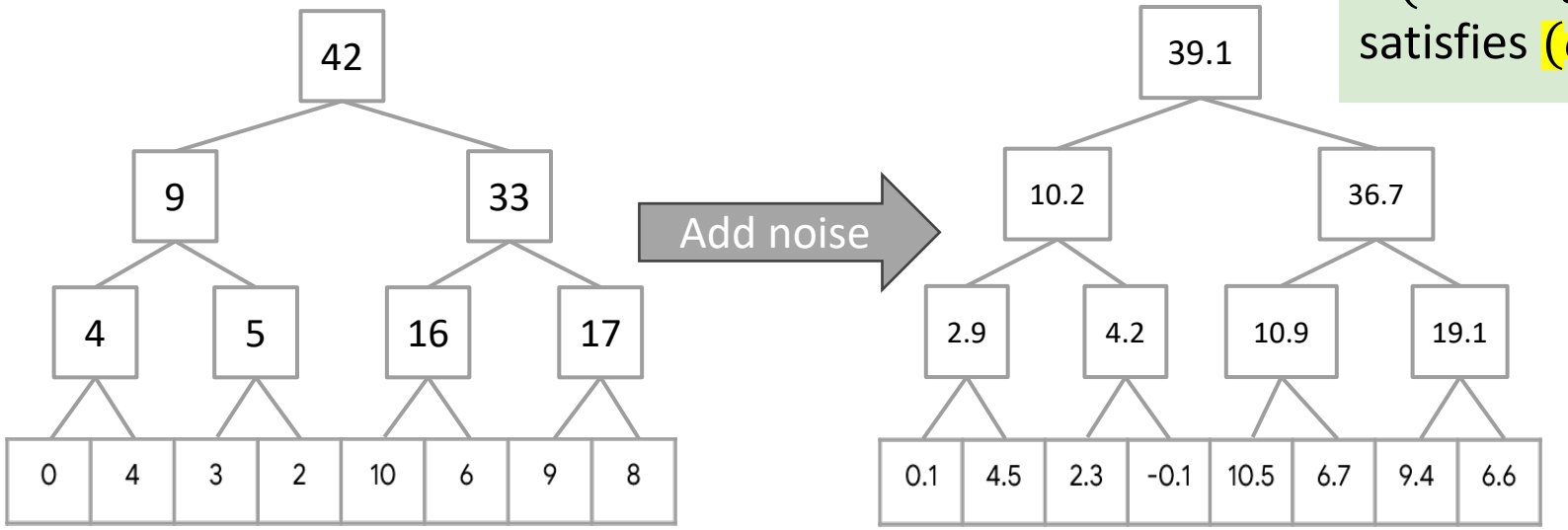
$\mathrm{Lap}(b) \propto e^{-x/b}$



Adding independent **Laplace noise** $\mathbf{Lap}(\boldsymbol{d}/\boldsymbol{\epsilon})$ to each query satisfies $\boldsymbol{\varepsilon}$-DP

Adding independent **Gaussian noise** $\mathbf{N}\left(\mathbf{0}, \boldsymbol{d}\,\mathbf{log}(\mathbf{1}/\boldsymbol{\delta})\,/\boldsymbol{\epsilon}^{\mathbf{2}}\right)$ to each query satisfies $(\boldsymbol{\epsilon}, \boldsymbol{\delta})$-DP

$\mathrm{N}(0, \sigma^2) \propto e^{-x^2/2\sigma^2}$



Results

Error / Privacy	Pure-DP $\delta = 0$	Approximate-DP $\delta \in (0, 1)$
Additive only $\alpha = 0$	$O(d/\epsilon)$ [Laplace Mechanism]	$O\left(\sqrt{d \log(1/\delta)}\,/\epsilon\right)$ [Gaussian Mechanism]
	$\Omega(d/\epsilon)$ [Our result]	$\Omega_{\epsilon,\delta}\left(\sqrt{d}\right)$ [Edmonds-Nikolov-Ullman'20]
Multiplicative & additive $\alpha \in (0, 1)$ is a constant	$\Omega(d/\epsilon)$ [Our result]	$O(\log(d/\delta)\,/\epsilon)$ [Our result]

Reduction to Classification

Reduction to Classification

Classification Problem

Reduction to Classification

Classification Problem

- Given threshold τ

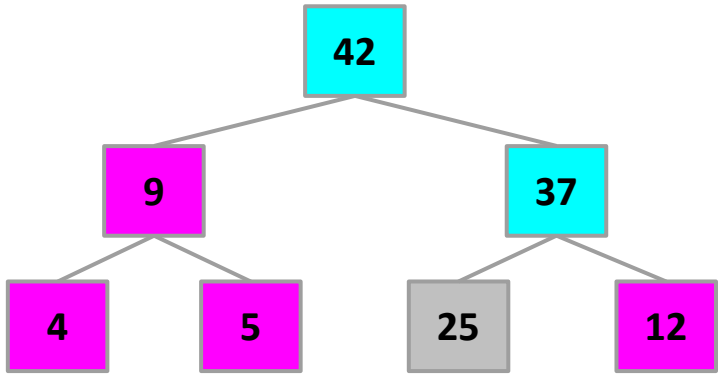
Reduction to Classification

Classification Problem

- Given threshold $\boldsymbol{\tau}$
- Output for each node $\boldsymbol{v}$ whether
  - $\boldsymbol{q_v}$ is above $\boldsymbol{(1+\alpha)\tau}$ or below $\boldsymbol{(1-\alpha)\tau}$
  - If $\boldsymbol{q_v} \in [\boldsymbol{(1-\alpha)\tau}, \boldsymbol{(1+\alpha)\tau}]$, any answer works

Reduction to Classification

Example: $\tau = 30, \alpha = 0.2$
Above: ≥ 36
Below: ≤ 24
Arbitrary: $24 \sim 36$

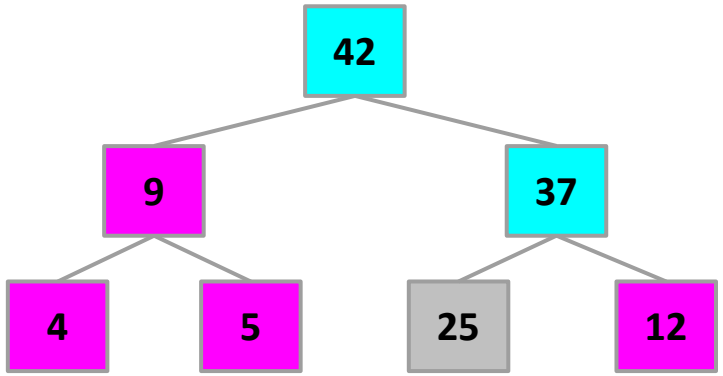


Classification Problem

- Given threshold τ
- Output for each node v whether
  - q_v is above $(1+\alpha)\tau$ or below $(1-\alpha)\tau$
  - If $q_v \in [(1-\alpha)\tau, (1+\alpha)\tau]$, any answer works

Reduction to Classification

Example: $\tau = 30, \alpha = 0.2$
Above: ≥ 36
Below: ≤ 24
Arbitrary: $24 \sim 36$



Classification Problem

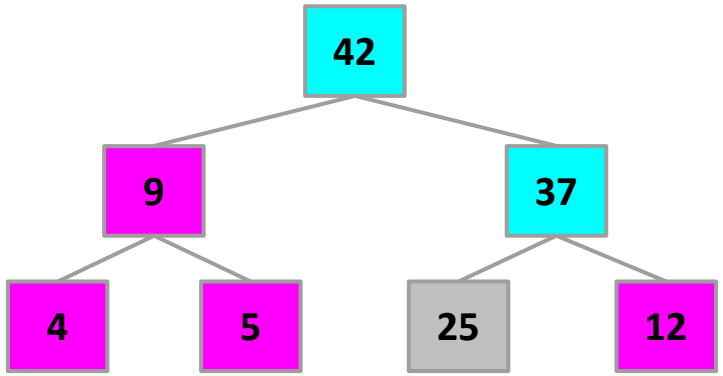
- Given threshold τ
- Output for each node v whether
  - q_v is above $(1+\alpha)\tau$ or below $(1-\alpha)\tau$
  - If $q_v \in [(1-\alpha)\tau, (1+\alpha)\tau]$, any answer works

Reduction

- Run *classification* on geometrically decreasing τ
- Use the smallest τ it is *above threshold* as estimate

Reduction to Classification

Example: $\tau = 30, \alpha = 0.2$
Above: ≥ 36
Below: ≤ 24
Arbitrary: $24 \sim 36$



Classification Problem

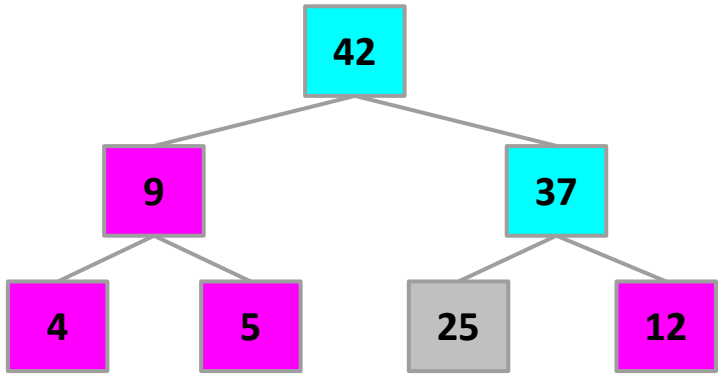
- Given threshold τ
- Output for each node v whether
  - q_v is above $(1+\alpha)\tau$ or below $(1-\alpha)\tau$
  - If $q_v \in [(1-\alpha)\tau, (1+\alpha)\tau]$, any answer works

Reduction

- Run *classification* on geometrically decreasing τ
  - Tracking privacy loss and error accumulation
- Use the smallest τ it is *above threshold* as estimate

Reduction to Classification

Example: $\tau = 30, \alpha = 0.2$
Above: ≥ 36
Below: ≤ 24
Arbitrary: $24 \sim 36$



Classification Problem

- Given threshold τ
- Output for each node v whether
  - q_v is above $(1 + \alpha)\tau$ and
  - If $q_v \in$ works

If the classification algorithm is private, the reduction result is also private

Reduction

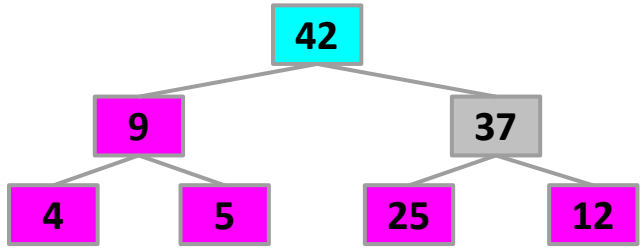
- Run *classification* on geometrically decreasing τ
  - Tracking privacy loss and error accumulation
- Use the smallest τ it is *above threshold* as estimate

Reduction Example

Reduction Example

$\tau = 35, \alpha = 0.2$

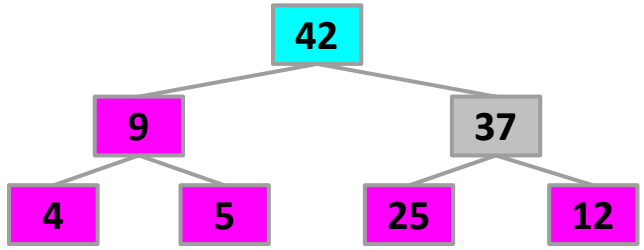
Above: ≥ 42 Below: ≤ 28



Reduction Example

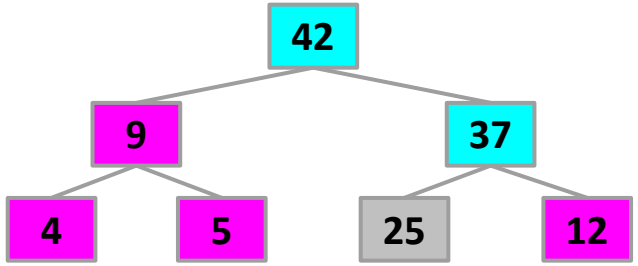
$\tau = 35, \alpha = 0.2$

Above: ≥ 42 Below: ≤ 28



$\tau = 25, \alpha = 0.2$

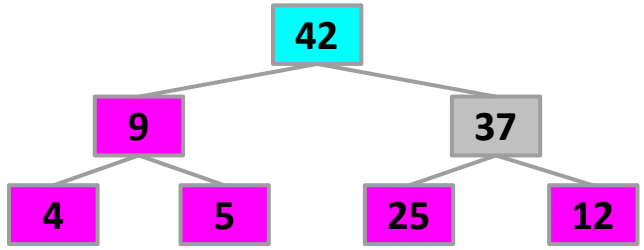
Above: ≥ 30 Below: ≤ 20



Reduction Example

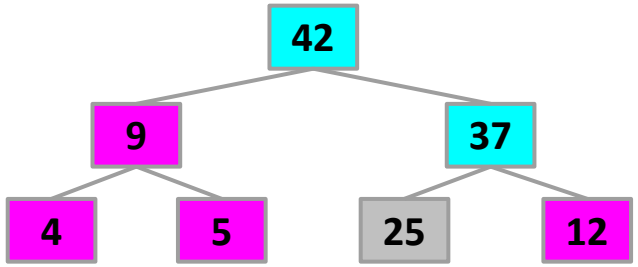
$\tau = 35, \alpha = 0.2$

Above: ≥ 42 Below: ≤ 28



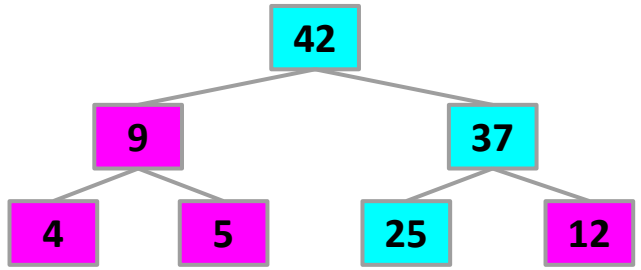
$\tau = 25, \alpha = 0.2$

Above: ≥ 30 Below: ≤ 20



$\tau = 15, \alpha = 0.2$

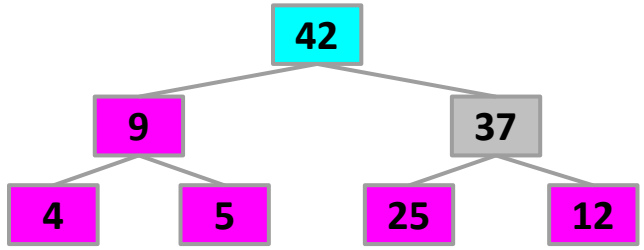
Above: ≥ 18 Below: ≤ 12



Reduction Example

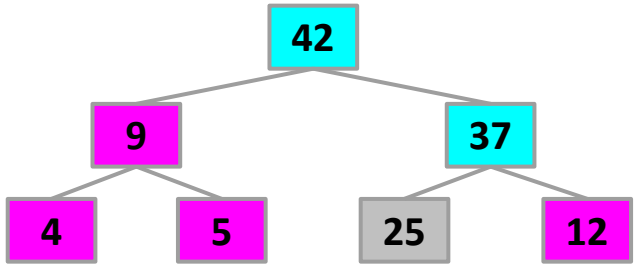
$\tau = 35, \alpha = 0.2$

Above: ≥ 42 Below: ≤ 28



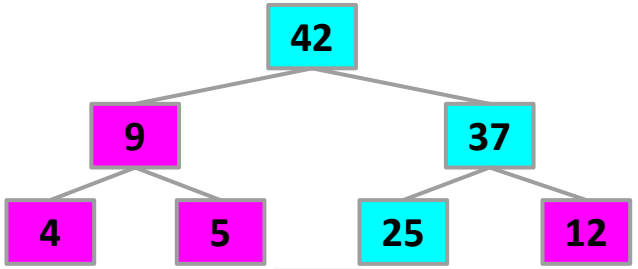
$\tau = 25, \alpha = 0.2$

Above: ≥ 30 Below: ≤ 20



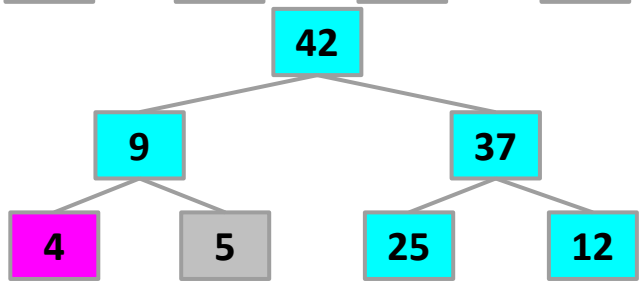
$\tau = 15, \alpha = 0.2$

Above: ≥ 18 Below: ≤ 12



$\tau = 5, \alpha = 0.2$

Above: ≥ 6 Below: ≤ 4



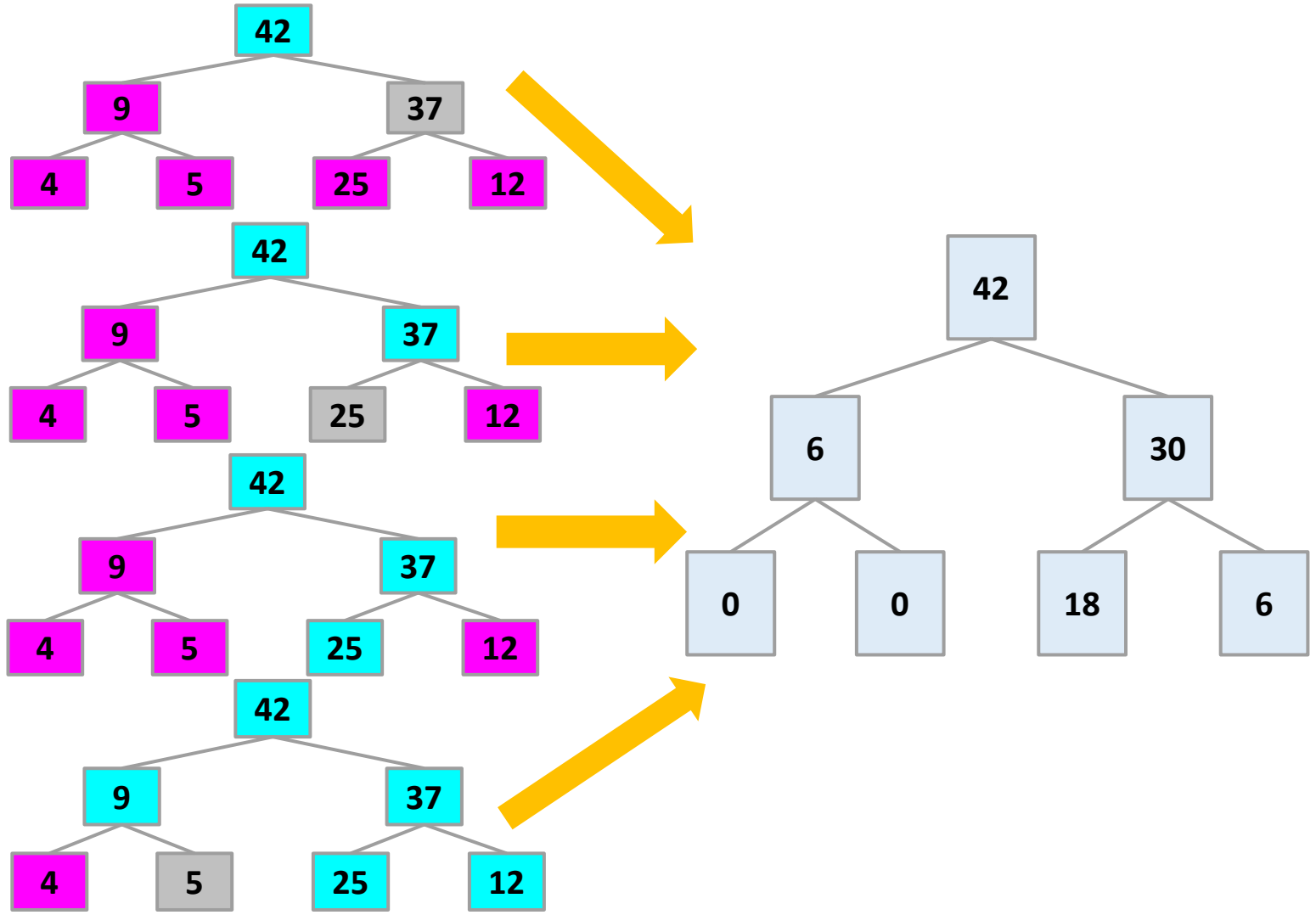
Reduction Example

$\tau = 35, \alpha = 0.2$
Above: ≥ 42 Below: ≤ 28

$\tau = 25, \alpha = 0.2$
Above: ≥ 30 Below: ≤ 20

$\tau = 15, \alpha = 0.2$
Above: ≥ 18 Below: ≤ 12

$\tau = 5, \alpha = 0.2$
Above: ≥ 6 Below: ≤ 4



Classification Algorithm

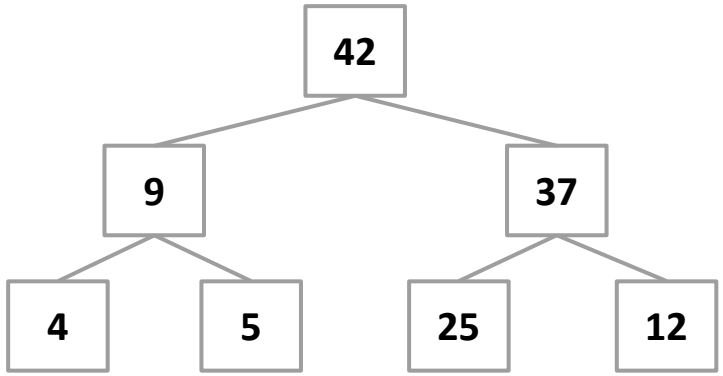
Classification Algorithm

Classification Problem

- Given threshold τ
- Promise that root count is $< 2\tau$
- Output for each node v whether
  - q_v is above $(1+\alpha)\tau$ or below $(1-\alpha)\tau$
  - If $q_v \in [(1-\alpha)\tau, (1+\alpha)\tau]$, any answer works

Classification Algorithm

Example: $\tau = 30$
Promise: root count is < 60

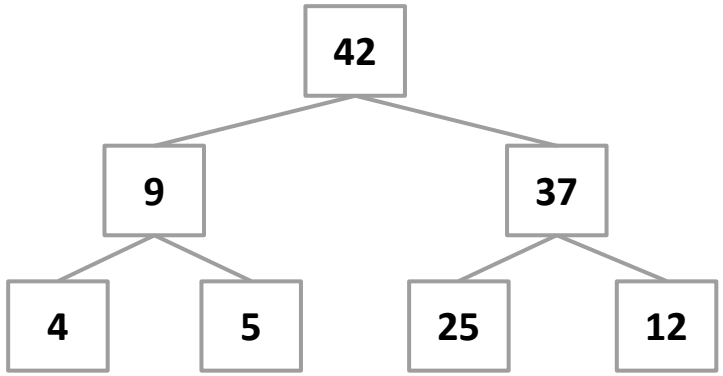


Classification Problem

- Given threshold τ
- Promise that root count is $< 2\tau$
- Output for each node v whether
  - q_v is above $(1+\alpha)\tau$ or below $(1-\alpha)\tau$
  - If $q_v \in [(1-\alpha)\tau, (1+\alpha)\tau]$, any answer works

Classification Algorithm

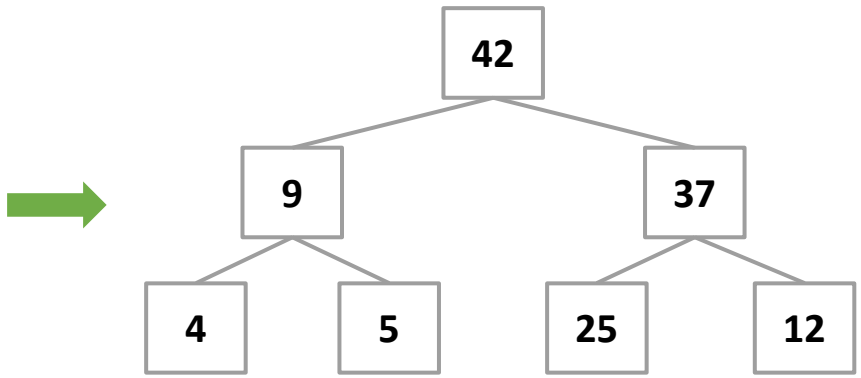
Example: $\tau = 30$
Promise: root count is < 60



Classification Algorithm

Classification Algorithm

Example: $\tau = 30$
Promise: root count is < 60

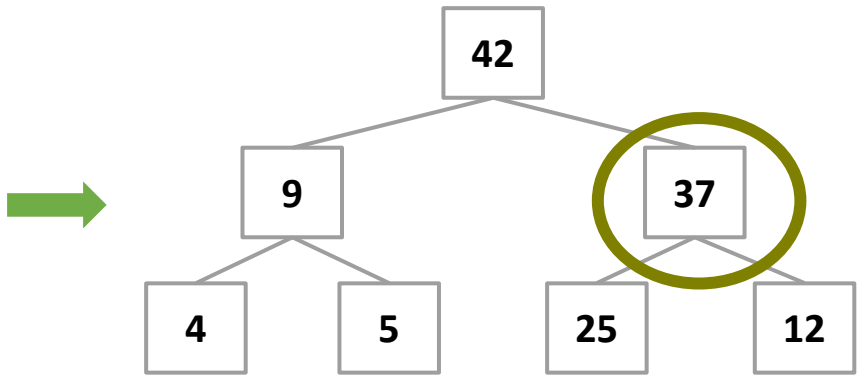


Classification Algorithm

- Select the *deepest* level containing a node with **noisy count** $\geq \tau$

Classification Algorithm

Example: $\tau = 30$
Promise: root count is < 60

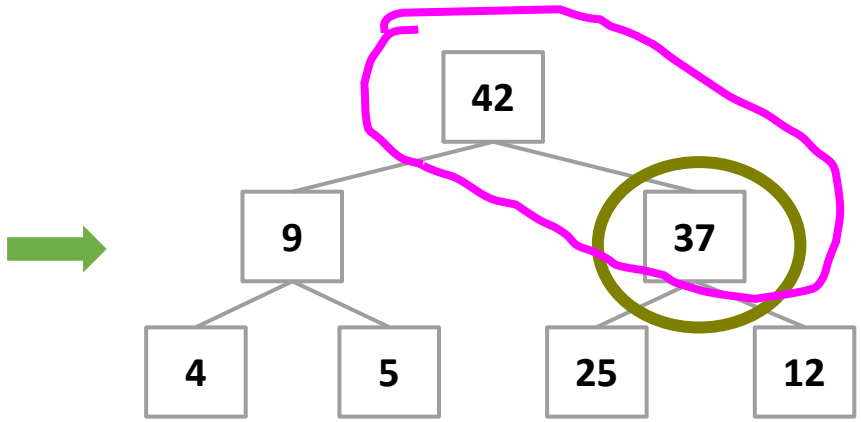


Classification Algorithm

- Select the *deepest* level containing a node with **noisy count** $\geq \tau$
- Select the nodes on that level with **noisy count** $\geq \tau$

Classification Algorithm

Example: $\tau = 30$
Promise: root count is < 60

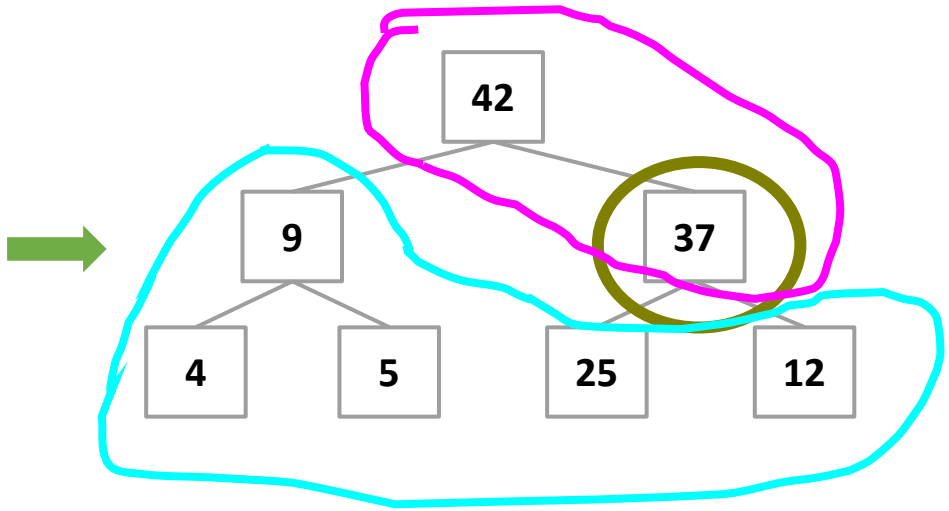


Classification Algorithm

- Select the *deepest* level containing a node with **noisy count** $\geq \tau$
- Select the nodes on that level with **noisy count** $\geq \tau$
- Mark all their ancestors as Above

Classification Algorithm

Example: $\tau = 30$
Promise: root count is < 60

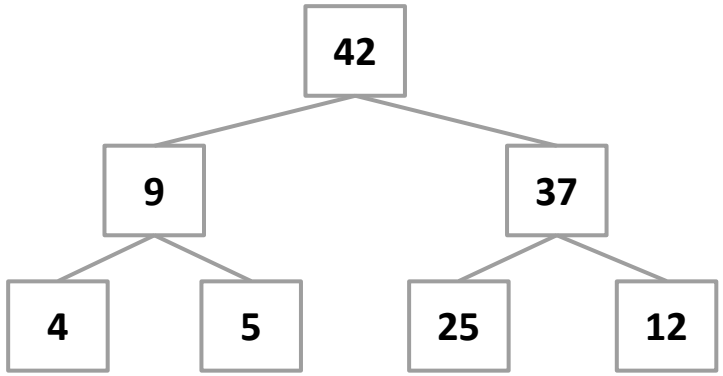


Classification Algorithm

- Select the *deepest* level containing a node with **noisy count** $\geq \tau$
- Select the nodes on that level with **noisy count** $\geq \tau$
- Mark all their ancestors as Above
- Mark all other nodes as Below

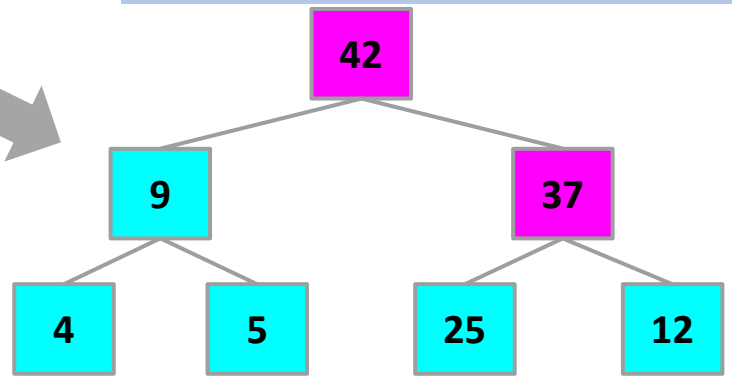
Classification Algorithm

Example: $\tau = 30$
Promise: root count is < 60



Classification Algorithm

- Select the *deepest* level containing a node with **noisy count** $\geq \tau$
- Select the nodes on that level with **noisy count** $\geq \tau$
- Mark all their ancestors as Above
- Mark all other nodes as Below



Ingredients

Classification Algorithm

- Select the *deepest* level containing a node with **noisy count** $\geq \tau$
- Select the nodes on that level with **noisy count** $\geq \tau$
- Mark all their ancestors as Above
- Mark all other nodes as Below

Ingredients

Classification Algorithm

- Select the *deepest* level containing a node with **noisy count** $\geq \tau$
- Select the nodes on that level with **noisy count** $\geq \tau$
- Mark all their ancestors as Above
- Mark all other nodes as Below

- Sparse vector technique [Dwork-Naor-Reingold-Rothblum-Vadhan'09]

Ingredients

Classification Algorithm

- Select the *deepest* level containing a node with **noisy count** $\geq \tau$
- Select the nodes on that level with **noisy count** $\geq \tau$
- Mark all their ancestors as Above
- Mark all other nodes as Below

- Sparse vector technique [Dwork-Naor-Reingold-Rothblum-Vadhan'09]
- ϵ-DP guarantee
- Error $O(\log(d)/\epsilon)$

Ingredients

Classification Algorithm

- Select the *deepest* level containing a node with **noisy count** $\geq \tau$
- Select the nodes on that level with **noisy count** $\geq \tau$
- Mark all their ancestors as Above
- Mark all other nodes as Below

- Sparse vector technique [Dwork-Naor-Reingold-Rothblum-Vadhan'09]
- ϵ-DP guarantee
- Error $O(\log(d)/\epsilon)$

Ingredients

Classification Algorithm

- Select the *deepest* level containing a node with **noisy count** $\geq \tau$
- Select the nodes on that level with **noisy count** $\geq \tau$
- Mark all their ancestors as Above
- Mark all other nodes as Below

- Sparse vector technique [Dwork-Naor-Reingold-Rothblum-Vadhan'09]
- ϵ-DP guarantee
- Error $O(\log(d)/\epsilon)$

- Truncated Laplace Mechanism [Geng-Ding-Guo-Kumar'20]

Ingredients

Classification Algorithm

- Select the *deepest* level containing a node with **noisy count** $\geq \tau$
- Select the nodes on that level with **noisy count** $\geq \tau$
- Mark all their ancestors as Above
- Mark all other nodes as Below

- Sparse vector technique [Dwork-Naor-Reingold-Rothblum-Vadhan'09]
- ϵ-DP guarantee
- Error $O(\log(d)/\epsilon)$

- Truncated Laplace Mechanism [Geng-Ding-Guo-Kumar'20]
- (ϵ, δ)-DP guarantee
- Error $O(\log(1/\delta)/\epsilon)$

Ingredients

Classification Algorithm

- Select the *deepest* level containing a node with **noisy count** $\geq \tau$
- Select the nodes on that level with **noisy count** $\geq \tau$
- Mark all their ancestors as Above
- Mark all other nodes as Below

- Sparse vector technique [Dwork-Naor-Reingold-Rothblum-Vadhan'09]
- ϵ-DP guarantee
- Error $O(\log(d)/\epsilon)$

- Truncated Laplace Mechanism [Geng-Ding-Guo-Kumar'20]
- (ϵ, δ)-DP guarantee
- Error $O(\log(1/\delta)/\epsilon)$

- Post-processing preserves privacy and accuracy

Ingredients

Classification Algorithm

- Select the *deepest* level containing a node with **noisy count** $\geq \tau$
- Select the nodes on that level with **noisy count** $\geq \tau$
- Mark all their ancestors as Above
- Mark all other nodes as Below

- Sparse vector technique [Dwork-Naor-Reingold-Rothblum-Vadhan'09]
- ϵ-DP guarantee
- Error $O(\log(d)/\epsilon)$

- Truncated Laplace Mechanism [Geng-Ding-Guo-Kumar'20]
- (ϵ, δ)-DP guarantee
- Error $O(\log(1/\delta)/\epsilon)$

- Post-processing preserves privacy and accuracy

Total Error: $O(\log(d/\delta)/\varepsilon)$

Results

Error / Privacy	Pure-DP $\delta = 0$	Approximate-DP $\delta \in (0, 1)$
Additive only $\alpha = 0$	$O(d/\epsilon)$ [Laplace Mechanism]	$O\left(\sqrt{d \log(1/\delta)}\,/\epsilon\right)$ [Gaussian Mechanism]
	$\Omega(d/\epsilon)$ [Our result]	$\Omega_{\epsilon,\delta}\left(\sqrt{d}\right)$ [Edmonds-Nikolov-Ullman'20]
Multiplicative & additive $\alpha \in (0, 1)$ is a constant	$\Omega(d/\epsilon)$ [Our result]	$O(\log(d/\delta)\,/\epsilon)$ [Our result]

Packing Argument [Hardt-Talwar'10]

Packing Argument [Hardt-Talwar'10]

- Let $\boldsymbol{A}$ be an $\boldsymbol{\epsilon}$-DP algorithm

Packing Argument [Hardt-Talwar'10]

- Let $\boldsymbol{A}$ be an $\boldsymbol{\epsilon}$-DP algorithm
- Construct hard inputs $\boldsymbol{x_1, x_2, \dots, x_m}$

Packing Argument [Hardt-Talwar'10]

- Let $\boldsymbol{A}$ be an $\boldsymbol{\epsilon}$-DP algorithm
- Construct hard inputs $\boldsymbol{x_1, x_2, \dots, x_m}$
- Design a decoding algorithm **Dec** which
  - Takes in $\boldsymbol{A(x_i)}$
  - Outputs an index $\boldsymbol{i' \in \{1, 2, \dots, m\}}$

Packing Argument [Hardt-Talwar'10]

- Let $\boldsymbol{A}$ be an $\boldsymbol{\epsilon}$-DP algorithm
- Construct hard inputs $\boldsymbol{x_1, x_2, \dots, x_m}$
- Design a decoding algorithm **Dec** which
  - Takes in $\boldsymbol{A(x_i)}$
  - Outputs an index $\boldsymbol{i' \in \{1, 2, \dots, m\}}$
- If $\boldsymbol{A}$ is too accurate,

Packing Argument [Hardt-Talwar'10]

- Let $\boldsymbol{A}$ be an $\boldsymbol{\epsilon}$-DP algorithm
- Construct hard inputs $\boldsymbol{x_1, x_2, \dots, x_m}$
- Design a decoding algorithm **Dec** which
  - Takes in $\boldsymbol{A(x_i)}$
  - Outputs an index $\boldsymbol{i' \in \{1, 2, \dots, m\}}$
- If $\boldsymbol{A}$ is too accurate,
  - $\Rightarrow$ **Dec** will get $\boldsymbol{i' = i}$ with high probability

Packing Argument [Hardt-Talwar'10]

- Let $\boldsymbol{A}$ be an $\boldsymbol{\epsilon}$-DP algorithm
- Construct hard inputs $\boldsymbol{x_1, x_2, \dots, x_m}$
- Design a decoding algorithm **Dec** which
  - Takes in $\boldsymbol{A(x_i)}$
  - Outputs an index $\boldsymbol{i' \in \{1, 2, \dots, m\}}$
- If $\boldsymbol{A}$ is too accurate,
  - $\Rightarrow$ **Dec** will get $\boldsymbol{i' = i}$ with high probability
  - $\Rightarrow$ violate privacy guarantee

Packing Argument [Hardt-Talwar'10]

- Let $\boldsymbol{A}$ be an $\boldsymbol{\epsilon}$-DP algorithm
- Construct hard inputs $\boldsymbol{x_1, x_2, \dots, x_m}$
- Design a decoding algorithm **Dec** which
  - Takes in $\boldsymbol{A(x_i)}$
  - Outputs an index $\boldsymbol{i' \in \{1, 2, \dots, m\}}$
- If $\boldsymbol{A}$ is too accurate,
  - $\Rightarrow$ **Dec** will get $\boldsymbol{i' = i}$ with high probability
  - $\Rightarrow$ violate privacy guarantee

Define $\boldsymbol{D = \max |x_i - x_j|_1}$

Packing Argument [Hardt-Talwar'10]

- Let A be an ϵ-DP algorithm
- Construct hard inputs $x_1, x_2, \dots, x_m$
- Design a decoding algorithm **Dec** which
  - Takes in $A(x_i)$
  - Outputs an index $i' \in \{1, 2, \dots, m\}$
- If A is too accurate,
  - $\Rightarrow$ **Dec** will get $i' = i$ with high probability
  - $\Rightarrow$ violate privacy guarantee

Define $D = \max |x_i - x_j|_1$

Assume for all i

$\Pr[\mathbf{Dec}(A(x_i)) = i] > \exp(\epsilon D)/m$

Packing Argument [Hardt-Talwar'10]

Define $\boldsymbol{D = \max |x_i - x_j|_1}$

- Let $\boldsymbol{A}$ be an $\boldsymbol{\epsilon}$-DP algorithm
- Construct hard inputs $\boldsymbol{x_1, x_2, \dots, x_m}$
- Design a decoding algorithm **Dec** which
  - Takes in $\boldsymbol{A(x_i)}$
  - Outputs an index $\boldsymbol{i' \in \{1, 2, \dots,}$
- If $\boldsymbol{A}$ is too accurate,
  - $\Rightarrow$ **Dec** will get $\boldsymbol{i' = i}$ with high
  - $\Rightarrow$ violate privacy guarantee

Assume for all $\boldsymbol{i}$

$$\boldsymbol{\Pr\left[\mathbf{Dec}\left(A(x_i)\right) = i\right] > \exp(\epsilon D)/m}$$

$$\begin{aligned} 1 &= \sum_{i'} \Pr\left[\mathrm{Dec}\left(A(x_1)\right) = i'\right] \\ &\geq \sum_{i'} \exp(-\epsilon D) \Pr\left[\mathrm{Dec}\left(A(x_{i'})\right) = i'\right] \\ &> \sum_{i'} \exp(-\epsilon D) \cdot \exp(\epsilon D)/m \\ &= 1 \end{aligned}$$

Packing Argument

ϵ-DP: If $|x - y|_1 = 1$, then for any event S

$$\Pr[A(x) \in S] \leq e^{\epsilon} \Pr[A(y) \in S]$$

- Let A be an ϵ-DP algorithm
- Construct hard inputs $x_1, x_2, \dots, x_m$
- Design a decoding algorithm **Dec** which
  - Takes in $A(x_i)$
  - Outputs an index $i' \in \{1, 2, \dots,$
- If A is too accurate,
  - $\Rightarrow$ **Dec** will get $i' = i$ with high
  - $\Rightarrow$ violate privacy guarantee

Define $D = \max |x_i - x_j|_1$

Assume for all i

$$\Pr\left[\mathbf{Dec}\left(A(x_i)\right) = i\right] > \exp(\epsilon D)/m$$

$$\begin{aligned} 1 &= \sum_{i'} \Pr\left[\mathrm{Dec}\left(A(x_1)\right) = i'\right] \\ &\geq \sum_{i'} \exp(-\epsilon D) \Pr\left[\mathrm{Dec}\left(A(x_{i'})\right) = i'\right] \\ &> \sum_{i'} \exp(-\epsilon D) \cdot \exp(\epsilon D)/m \\ &= 1 \end{aligned}$$

Packing Argument

ϵ-DP: If $|x - y|_1 = 1$, then for any event S

$$\Pr[A(x) \in S] \leq e^{\epsilon} \Pr[A(y) \in S]$$

- Let A be an ϵ-DP algorithm
- Construct hard inputs $x_1, x_2, \dots, x_m$
- Design a decoding algorithm **Dec** which
  - Takes in $A(x_i)$
  - Outputs an index $i' \in \{1, 2, \dots,$
- If A is too accurate,
  - $\Rightarrow$ **Dec** will get $i' = i$ with high
  - $\Rightarrow$ violate privacy guarantee

Define $D = \max |x_i - x_j|_1$

Assume for all i

$$\Pr[\mathbf{Dec}(A(x_i)) = i] > \exp(\epsilon D)/m$$

$$\begin{aligned} 1 &= \sum_{i'} \Pr[\mathrm{Dec}(A(x_1)) = i'] \\ &\geq \sum_{i'} \exp(-\epsilon D) \Pr\left[\mathrm{Dec}\left(A(x_{i'})\right) = i'\right] \\ &> \sum_{i'} \exp(-\epsilon D) \cdot \exp(\epsilon D)/m \\ &= 1 \end{aligned}$$

Hard Inputs

- Let $\boldsymbol{A}$ be an $\boldsymbol{\epsilon}$-DP algorithm
- Construct hard inputs $\boldsymbol{x_1, x_2, \dots, x_m}$
  - Pairwise distance bounded by $\boldsymbol{D}$
- Design a decoding algorithm **Dec** which
  - Takes in $\boldsymbol{A(x_i)}$
  - Outputs an index $\boldsymbol{i' \in \{1, 2, \dots, m\}}$
  - Satisfies $\boldsymbol{\mathbf{Pr}[i' = i] > \mathbf{exp}(\epsilon D)/m}$

Hard Inputs

- Let $\boldsymbol{A}$ be an $\boldsymbol{\epsilon}$-DP algorithm
- Construct hard inputs $\boldsymbol{x_1, x_2, \dots, x_m}$
  - Pairwise distance bounded by $\boldsymbol{D}$
- Design a decoding algorithm **Dec** which
  - Takes in $\boldsymbol{A(x_i)}$
  - Outputs an index $\boldsymbol{i' \in \{1, 2, \dots, m\}}$
  - Satisfies $\boldsymbol{\mathbf{Pr}[i' = i] > \mathbf{exp}(\epsilon D)/m}$

Hard Inputs

- Let $\boldsymbol{A}$ be an $\boldsymbol{\epsilon}$-DP algorithm
- Construct hard inputs $\boldsymbol{x_1, x_2, \dots, x_m}$
  - Pairwise distance bounded by $\boldsymbol{D}$
- Design a decoding algorithm **Dec** which
  - Takes in $\boldsymbol{A(x_i)}$
  - Outputs an index $\boldsymbol{i' \in \{1, 2, \dots, m\}}$
  - Satisfies $\boldsymbol{\mathbf{Pr}[i' = i] > \mathbf{exp}(\epsilon D)/m}$

$\boldsymbol{x_i}$: put $\boldsymbol{0.1d/\epsilon}$ at the $\boldsymbol{i}$-th leaf & **zero** elsewhere

$\boldsymbol{m = 2^d}$ and $\boldsymbol{D = 0.2d/\epsilon}$

Hard Inputs

- Let $\boldsymbol{A}$ be an $\boldsymbol{\epsilon}$-DP algorithm
- Construct hard inputs $\boldsymbol{x_1, x_2, \dots, x_m}$
  - Pairwise distance bounded by $\boldsymbol{D}$
- Design a decoding algorithm **Dec** which
  - Takes in $\boldsymbol{A(x_i)}$
  - Outputs an index $\boldsymbol{i' \in \{1, 2, \dots, m\}}$
  - Satisfies $\boldsymbol{\Pr[i' = i] > \exp(\epsilon D)/m}$

$\boldsymbol{x_i}$: put $\boldsymbol{0.1d/\epsilon}$ at the $\boldsymbol{i}$-th leaf & **zero** elsewhere

$\boldsymbol{m = 2^d}$ and $\boldsymbol{D = 0.2d/\epsilon}$

Example: $\boldsymbol{i = 5, d = 3, \epsilon = 1}$

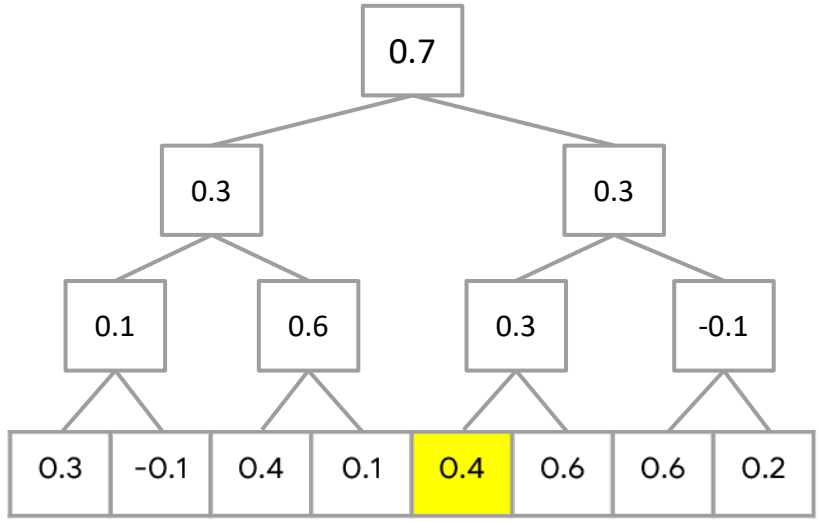
Hard Inputs

- Let $\boldsymbol{A}$ be an $\boldsymbol{\epsilon}$-DP algorithm
- Construct hard inputs $\boldsymbol{x_1, x_2, \dots, x_m}$
  - Pairwise distance bounded by $\boldsymbol{D}$
- Design a decoding algorithm **Dec** which
  - Takes in $\boldsymbol{A(x_i)}$
  - Outputs an index $\boldsymbol{i' \in \{1, 2, \dots, m\}}$
  - Satisfies $\boldsymbol{\Pr[i' = i] > \exp(\epsilon D)/m}$

$\boldsymbol{x_i}$: put $\boldsymbol{0.1d/\epsilon}$ at the $\boldsymbol{i}$-th leaf & **zero** elsewhere

$\boldsymbol{m = 2^d}$ and $\boldsymbol{D = 0.2d/\epsilon}$

Output of $\boldsymbol{A(xi)}$



Example: $\boldsymbol{i = 5, d = 3, \epsilon = 1}$

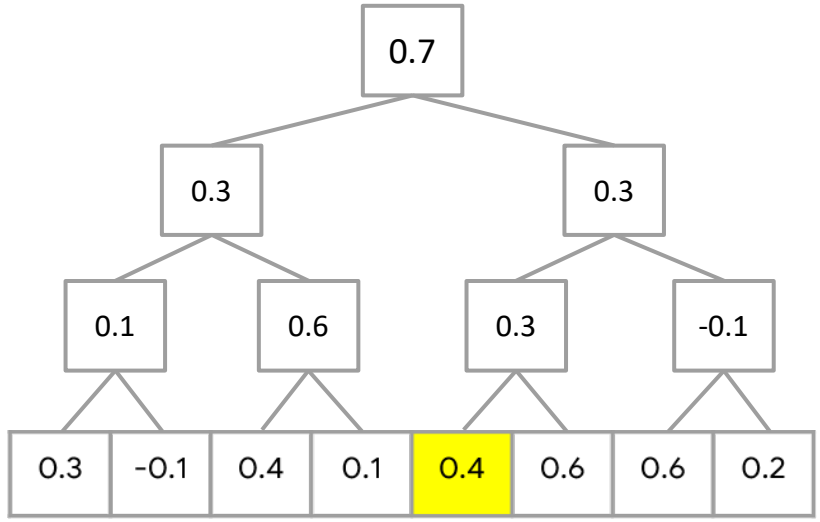
Hard Inputs

- Let $\boldsymbol{A}$ be an $\boldsymbol{\epsilon}$-DP algorithm
- Construct hard inputs $\boldsymbol{x_1, x_2, \dots, x_m}$
  - Pairwise distance bounded by $\boldsymbol{D}$
- Design a decoding algorithm **Dec** which
  - Takes in $\boldsymbol{A(x_i)}$
  - Outputs an index $\boldsymbol{i' \in \{1, 2, \dots, m\}}$
  - Satisfies $\boldsymbol{\mathbf{Pr}[i' = i] > \mathbf{exp}(\epsilon D)/m}$

$\boldsymbol{x_i}$: put $\boldsymbol{0.1d/\epsilon}$ at the $\boldsymbol{i}$-th leaf & **zero** elsewhere

$\boldsymbol{m = 2^d}$ and $\boldsymbol{D = 0.2d/\epsilon}$

Output of $\boldsymbol{A(xi)}$



Example: $\boldsymbol{i = 5, d = 3, \epsilon = 1}$

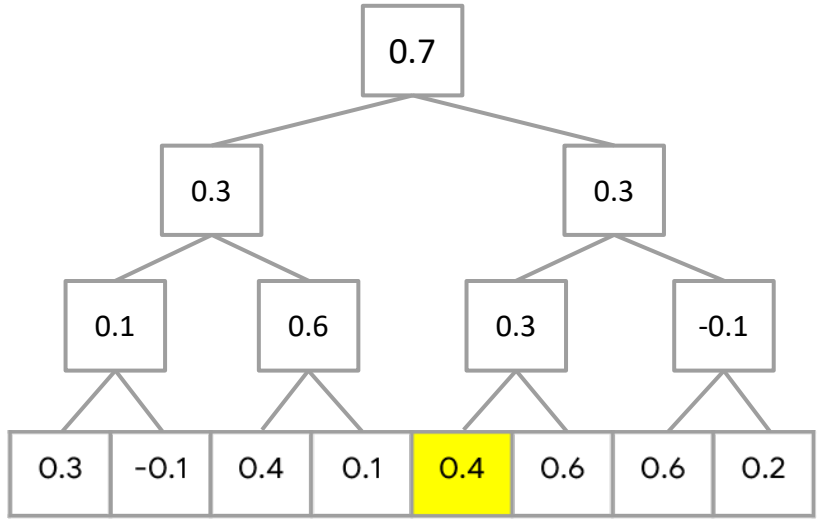
Hard Inputs

- Let $\boldsymbol{A}$ be an $\boldsymbol{\epsilon}$-DP algorithm
- Construct hard inputs $\boldsymbol{x_1, x_2, \dots, x_m}$
  - Pairwise distance bounded by $\boldsymbol{D}$
- Design a decoding algorithm **Dec** which
  - Takes in $\boldsymbol{A(x_i)}$
  - Outputs an index $\boldsymbol{i' \in \{1, 2, \dots, m\}}$
  - Satisfies $\boldsymbol{\Pr[i' = i] > \exp(\epsilon D)/m}$

$\boldsymbol{x_i}$: put $\boldsymbol{0.1d/\epsilon}$ at the $\boldsymbol{i}$-th leaf & **zero** elsewhere
$\boldsymbol{m = 2^d}$ and $\boldsymbol{D = 0.2d/\epsilon}$

$\boldsymbol{\Pr[i' = i] > \exp(0.2d)/2^d \approx 0.6^d}$

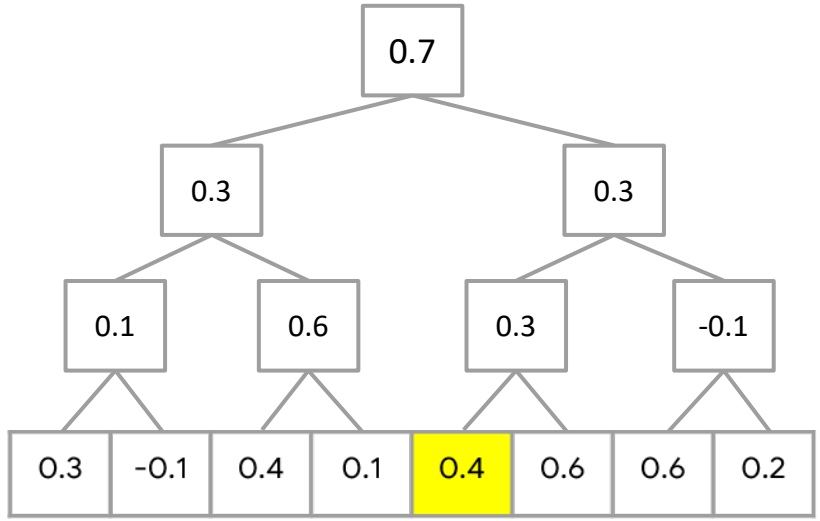
Output of $\boldsymbol{A(xi)}$



Example: $\boldsymbol{i = 5, d = 3, \epsilon = 1}$

Decoding Algorithm

Output of $A(xi)$



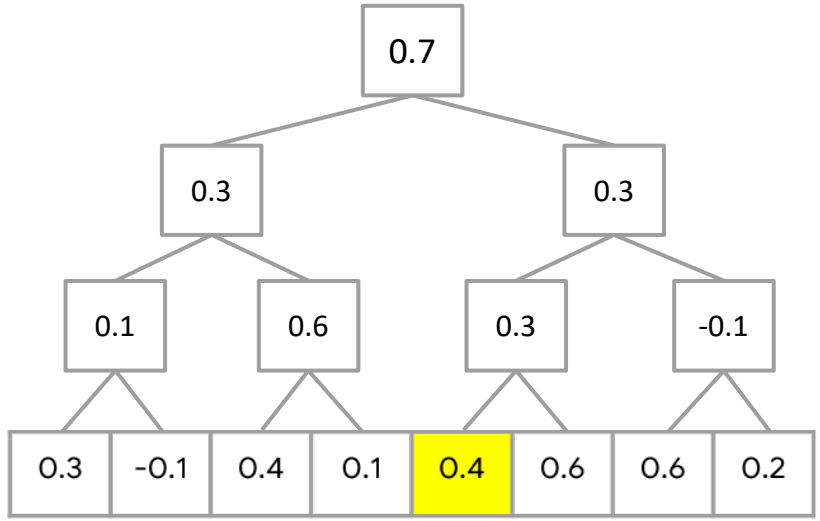
Goal: $\mathbf{Pr}[i' = i] > 0.6^d = 0.216$

Example: $i = 5, d = 3, \epsilon = 1$
$0.1d/\epsilon = 0.3$

Decoding Algorithm

- Start at the root and walk down to a leaf i'

Output of $A(xi)$



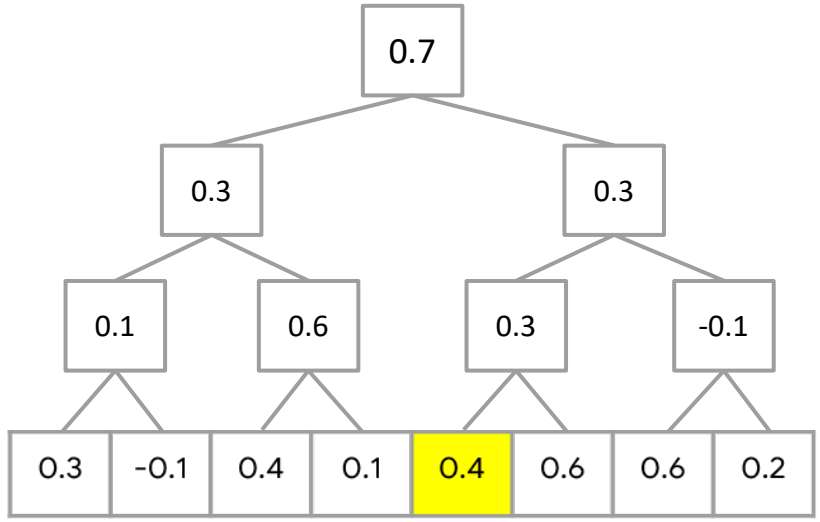
Goal: $\mathbf{Pr}[i' = i] > 0.6^d = 0.216$

Example: $i = 5, d = 3, \epsilon = 1$
$0.1d/\epsilon = 0.3$

Decoding Algorithm

- Start at the root and walk down to a leaf i'
- Consider current node's children

Output of $A(xi)$



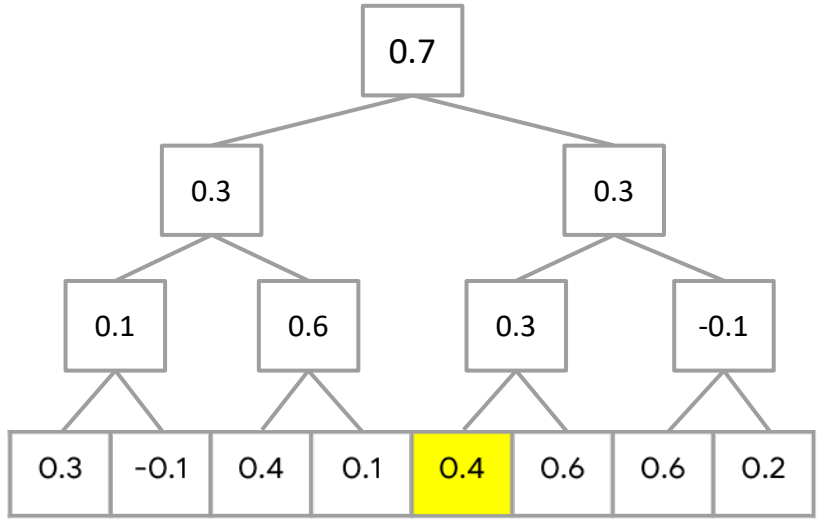
Goal: $\mathbf{Pr}[i' = i] > 0.6^d = 0.216$

Example: $i = 5, d = 3, \epsilon = 1$
$0.1d/\epsilon = 0.3$

Decoding Algorithm

- Start at the root and walk down to a leaf i'
- Consider current node's children
  - If only one of them is $\sim 0.1d/\epsilon$
    - Move to that child w.p. 0.9
    - Move to the other one w.p. 0.1

Output of $A(xi)$



Goal: $\mathbf{Pr}[i' = i] > 0.6^d = 0.216$

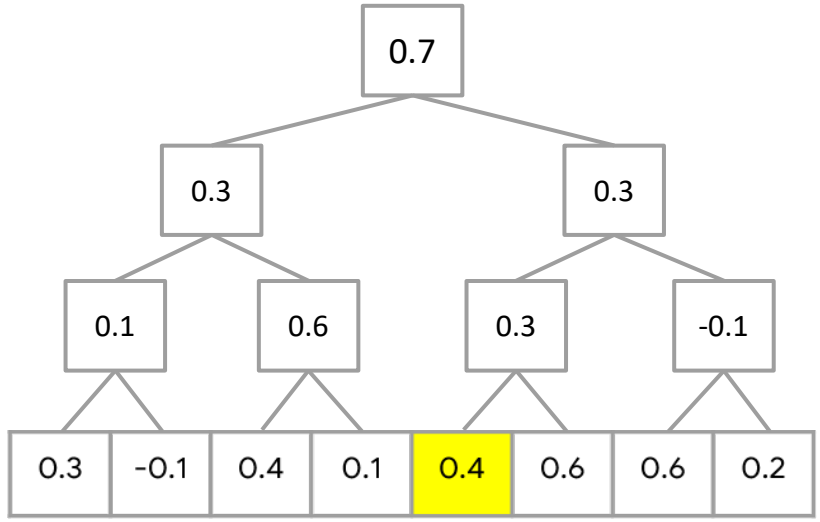
Example: $i = 5, d = 3, \epsilon = 1$
$0.1d/\epsilon = 0.3$

Decoding Algorithm

- Start at the root and walk down to a leaf i'
- Consider current node's children
  - If only one of them is $\sim 0.1d/\epsilon$
    - Move to that child w.p. 0.9
    - Move to the other one w.p. 0.1
  - Otherwise, move to each child w.p. 0.5

Goal: $\mathbf{Pr}[i' = i] > 0.6^d = 0.216$

Output of $A(xi)$



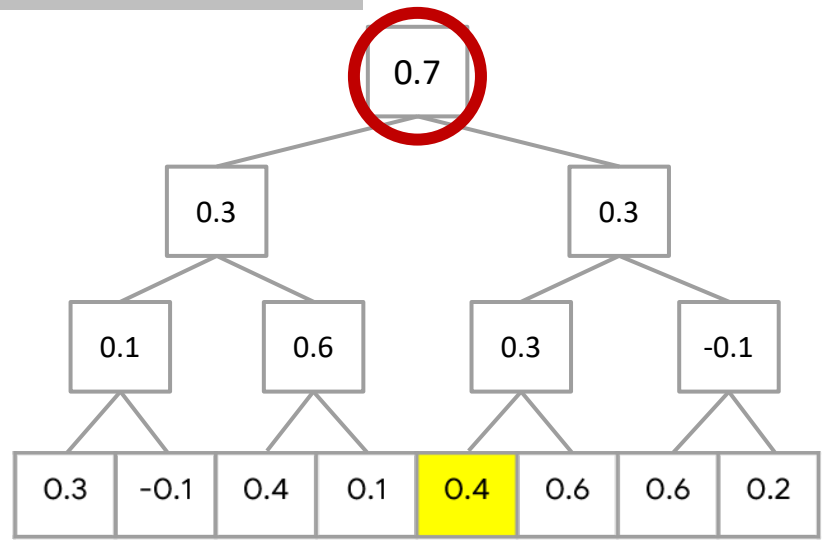
Example: $i = 5, d = 3, \epsilon = 1$
$0.1d/\epsilon = 0.3$

Decoding Algorithm

- Start at the root and walk down to a leaf i'
- Consider current node's children
  - If only one of them is $\sim 0.1d/\epsilon$
    - Move to that child w.p. 0.9
    - Move to the other one w.p. 0.1
  - Otherwise, move to each child w.p. 0.5

Goal: $\mathbf{Pr}[i' = i] > 0.6^d = 0.216$

Output of $A(xi)$



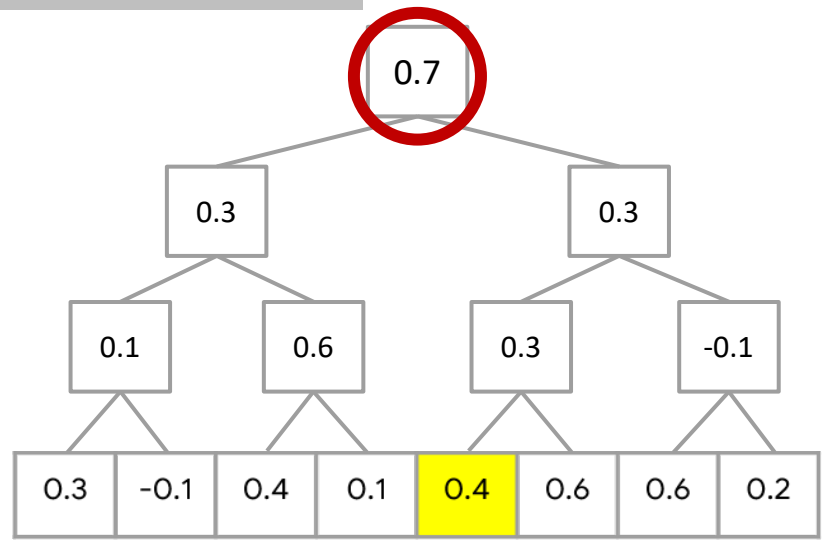
Example: $i = 5, d = 3, \epsilon = 1$
$0.1d/\epsilon = 0.3$

Decoding Algorithm

- Start at the root and walk down to a leaf i'
- Consider current node's children
  - If only one of them is $\sim 0.1d/\epsilon$
    - Move to that child w.p. 0.9
    - Move to the other one w.p. 0.1
  - Otherwise, move to each child w.p. 0.5

Goal: $\mathbf{Pr}[i' = i] > 0.6^d = 0.216$

Output of $A(xi)$



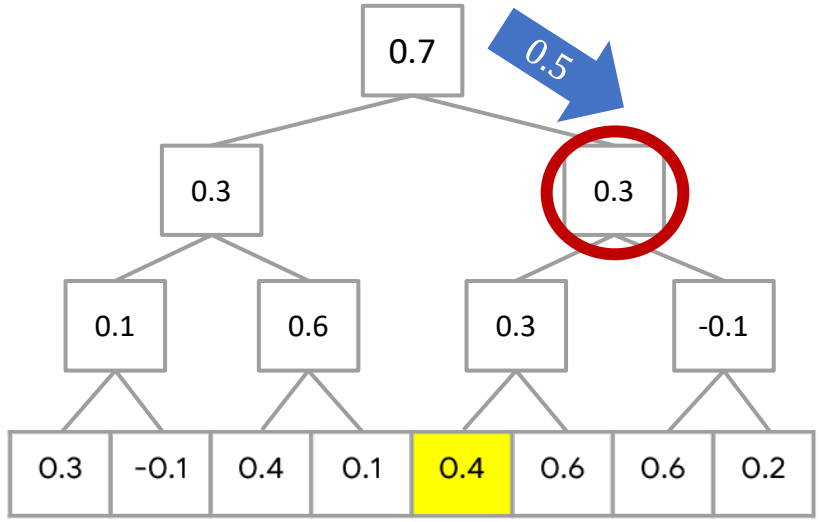
Example: $i = 5, d = 3, \epsilon = 1$
$0.1d/\epsilon = 0.3$

Decoding Algorithm

- Start at the root and walk down to a leaf i'
- Consider current node's children
  - If only one of them is $\sim 0.1d/\epsilon$
    - Move to that child w.p. 0.9
    - Move to the other one w.p. 0.1
  - Otherwise, move to each child w.p. 0.5

Goal: $\mathbf{Pr}[i' = i] > 0.6^d = 0.216$

Output of $A(xi)$



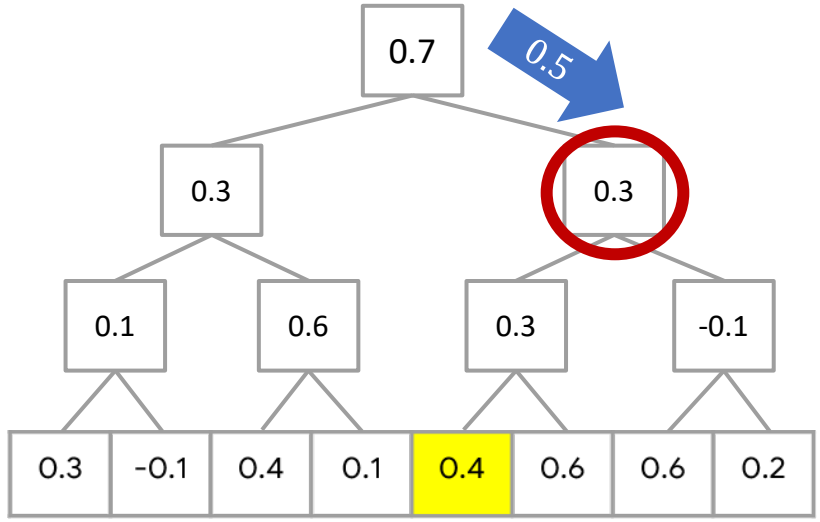
Example: $i = 5, d = 3, \epsilon = 1$
$0.1d/\epsilon = 0.3$

Decoding Algorithm

- Start at the root and walk down to a leaf i'
- Consider current node's children
  - If only one of them is $\sim 0.1d/\epsilon$
    - Move to that child w.p. 0.9
    - Move to the other one w.p. 0.1
  - Otherwise, move to each child w.p. 0.5

Goal: $\mathbf{Pr}[i' = i] > 0.6^d = 0.216$

Output of $A(xi)$



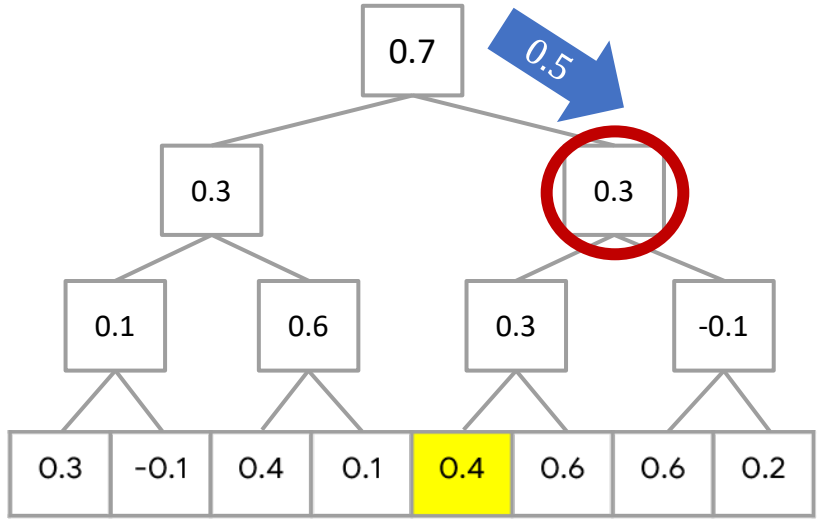
Example: $i = 5, d = 3, \epsilon = 1$
$0.1d/\epsilon = 0.3$

Decoding Algorithm

- Start at the root and walk down to a leaf i'
- Consider current node's children
  - If only one of them is $\sim 0.1d/\epsilon$
    - Move to that child w.p. 0.9
    - Move to the other one w.p. 0.1
  - Otherwise, move to each child w.p. 0.5

Goal: $\mathbf{Pr}[i' = i] > 0.6^d = 0.216$

Output of $A(xi)$



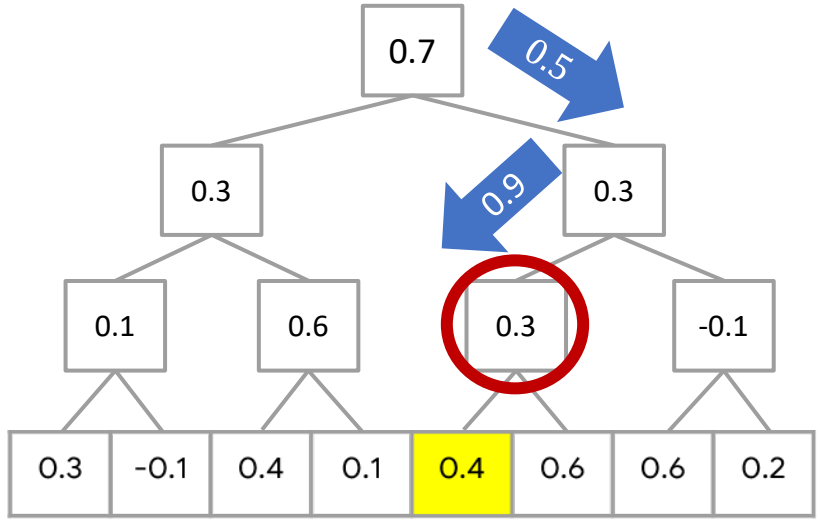
Example: $i = 5, d = 3, \epsilon = 1$
$0.1d/\epsilon = 0.3$

Decoding Algorithm

- Start at the root and walk down to a leaf i'
- Consider current node's children
  - If only one of them is $\sim 0.1d/\epsilon$
    - Move to that child w.p. 0.9
    - Move to the other one w.p. 0.1
  - Otherwise, move to each child w.p. 0.5

Goal: $\mathbf{Pr}[i' = i] > 0.6^d = 0.216$

Output of $A(xi)$



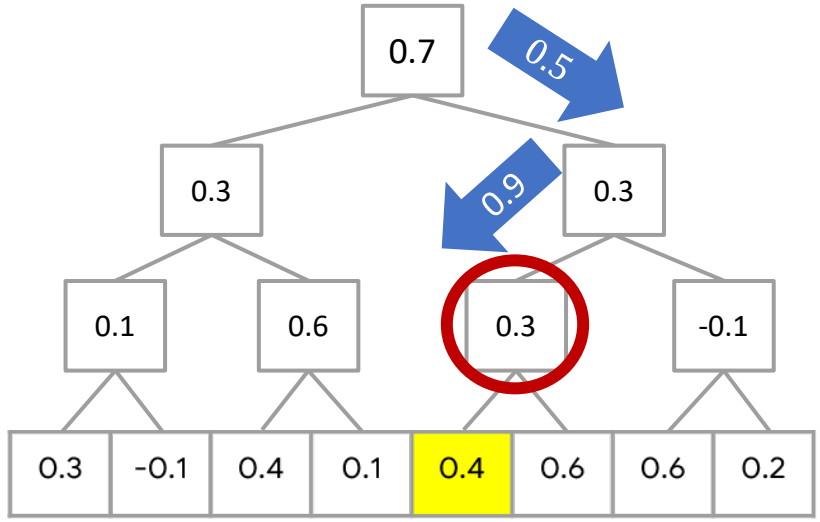
Example: $i = 5, d = 3, \epsilon = 1$
$0.1d/\epsilon = 0.3$

Decoding Algorithm

- Start at the root and walk down to a leaf i'
- Consider current node's children
  - If only one of them is $\sim 0.1d/\epsilon$
    - Move to that child w.p. 0.9
    - Move to the other one w.p. 0.1
  - Otherwise, move to each child w.p. 0.5

Goal: $\mathbf{Pr}[i' = i] > 0.6^d = 0.216$

Output of $A(xi)$



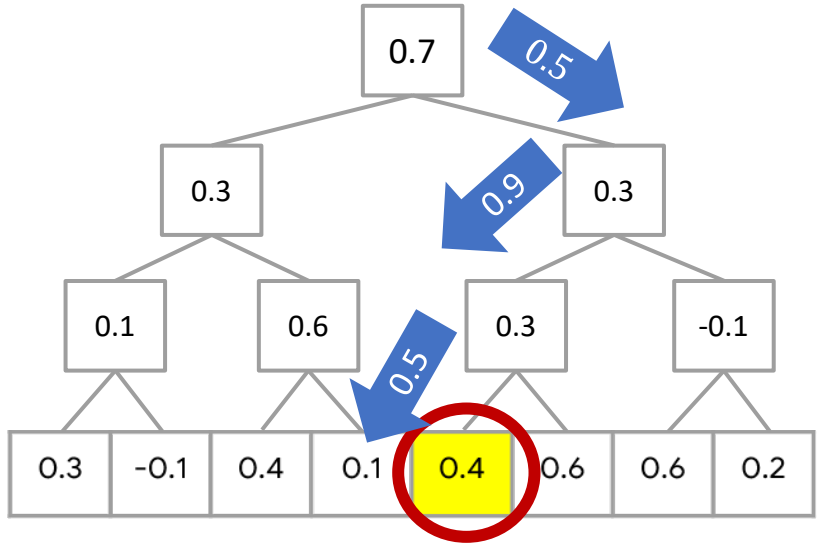
Example: $i = 5, d = 3, \epsilon = 1$
$0.1d/\epsilon = 0.3$

Decoding Algorithm

- Start at the root and walk down to a leaf i'
- Consider current node's children
  - If only one of them is $\sim 0.1d/\epsilon$
    - Move to that child w.p. 0.9
    - Move to the other one w.p. 0.1
  - Otherwise, move to each child w.p. 0.5

Goal: $\mathbf{Pr}[i' = i] > 0.6^d = 0.216$

Output of $A(xi)$



Example: $i = 5, d = 3, \epsilon = 1$
$0.1d/\epsilon = 0.3$

Decoding Algorithm

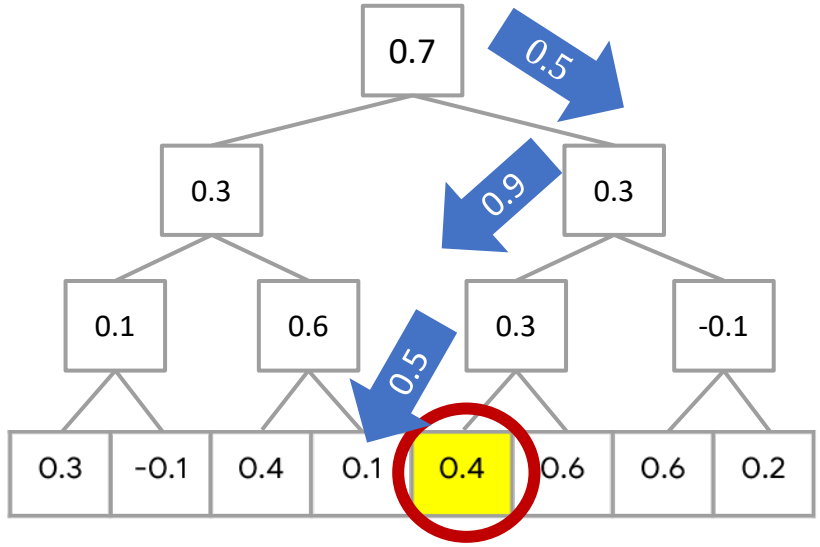
- Start at the root and walk down to a leaf i'
- Consider current node's children
  - If only one of them is $\sim 0.1d/\epsilon$
    - Move to that child w.p. 0.9
    - Move to the other one w.p. 0.1
  - Otherwise, move to each child w.p. 0.5

Goal: $\mathbf{Pr}[i' = i] > 0.6^d = 0.216$

In this example,

$$\mathbf{Pr}[i' = i] = 0.5 \cdot 0.9 \cdot 0.5 = 0.225$$

Output of $A(xi)$



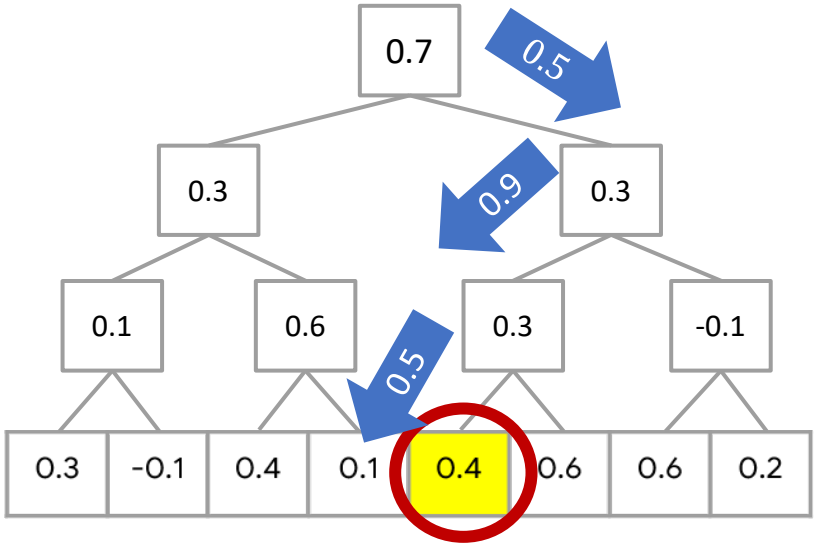
Example: $i = 5, d = 3, \epsilon = 1$

$0.1d/\epsilon = 0.3$

Decoding Algorithm

- Start at the root and walk down to a leaf i'
- Consider current node's children
  - If only one of them is $\sim 0.1d/\epsilon$
    - Move to that child w.p. 0.9
    - Move to the other one w.p. 0.1
  - Otherwise, move to each child w.p. 0.5

Output of $\boldsymbol{A(xi)}$



Goal: $\mathbf{Pr}[\boldsymbol{i'} = \boldsymbol{i}] > \mathbf{0.6}^{\boldsymbol{d}} = \mathbf{0.216}$

In this example,

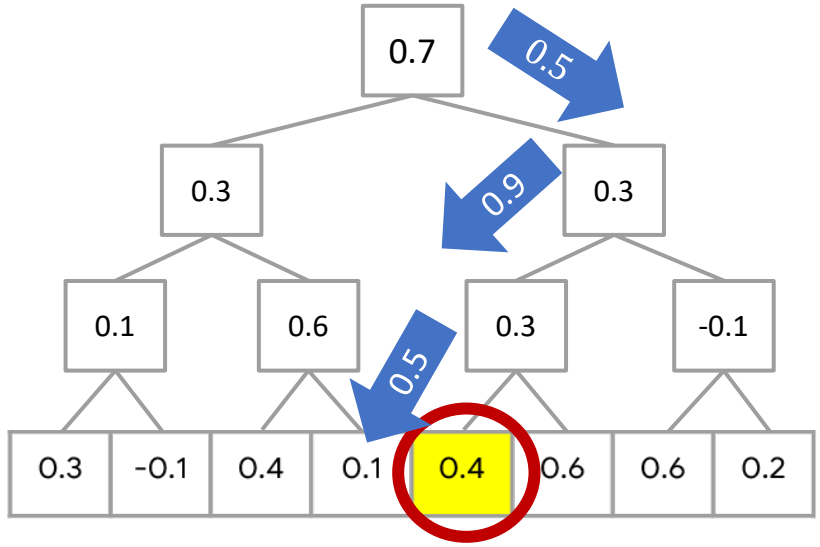
$\mathbf{Pr}[\boldsymbol{i'} = \boldsymbol{i}] = \mathbf{0.5} \cdot \mathbf{0.9} \cdot \mathbf{0.5} = \mathbf{0.225}$

In general, if the **error is small**,

Decoding Algorithm

- Start at the root and walk down to a leaf i'
- Consider current node's children
  - If only one of them is $\sim 0.1d/\epsilon$
    - Move to that child w.p. 0.9
    - Move to the other one w.p. 0.1
  - Otherwise, move to each child w.p. 0.5

Output of $A(xi)$



Goal: $\mathbf{Pr}[i' = i] > 0.6^d = 0.216$

In this example,

$\mathbf{Pr}[i' = i] = 0.5 \cdot 0.9 \cdot 0.5 = 0.225$

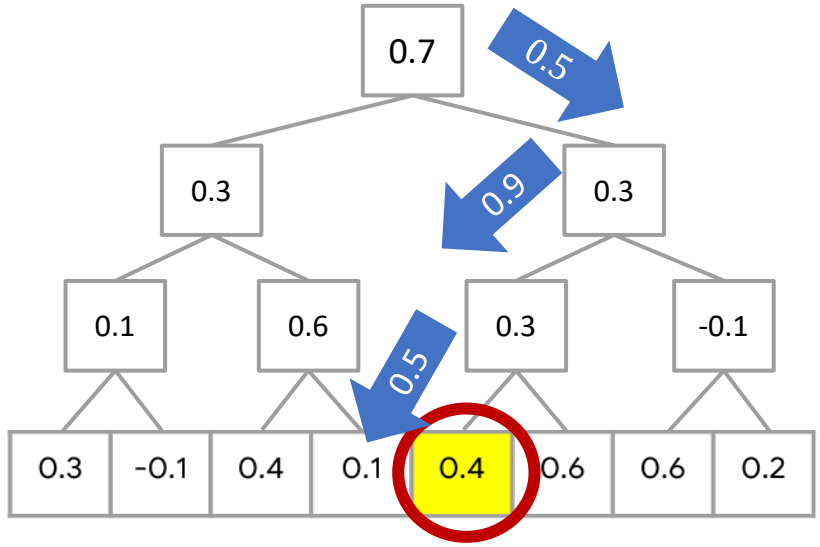
In general, if the **error is small**,

- $\Rightarrow$ in this case for **most steps**

Decoding Algorithm

- Start at the root and walk down to a leaf i'
- Consider current node's children
  - If only one of them is $\sim 0.1d/\epsilon$
    - Move to that child w.p. 0.9
    - Move to the other one w.p. 0.1
  - Otherwise, move to each child w.p. 0.5

Output of $A(xi)$



Goal: $\mathbf{Pr}[i' = i] > 0.6^d = 0.216$

In this example,
$\mathbf{Pr}[i' = i] = 0.5 \cdot 0.9 \cdot 0.5 = 0.225$

In general, if the **error is small**,
- $\Rightarrow$ in this case for **most steps**
- $\Rightarrow$ biased towards the correct leaf

Conclusion

Conclusion

- Counting tasks on trees captures real-life hierarchical problems
  - Census, Web Ads, ...

Conclusion

- Counting tasks on trees captures real-life hierarchical problems
  - Census, Web Ads, ...
- **Error metric**: additive-only ⇒ multiplicative-additive
  - Heuristically and rigorously justified

Conclusion

- Counting tasks on trees captures real-life hierarchical problems
  - Census, Web Ads, ...
- **Error metric**: additive-only ⇒ multiplicative-additive
  - Heuristically and rigorously justified
- Pure-DP protocol and a matching lower bound

Conclusion

- Counting tasks on trees captures real-life hierarchical problems
  - Census, Web Ads, ...
- **Error metric**: additive-only ⇒ multiplicative-additive
  - Heuristically and rigorously justified
- Pure-DP protocol and a matching lower bound
- Approximate-DP protocol

Conclusion

- Counting tasks on trees captures real-life hierarchical problems
  - Census, Web Ads, ...
- **Error metric**: additive-only ⇒ multiplicative-additive
  - Heuristically and rigorously justified
- Pure-DP protocol and a matching lower bound
- Approximate-DP protocol
  - Open directions: A matching lower bound? Or a better algorithm?

Conclusion

- Counting tasks on trees captures real-life hierarchical problems
  - Census, Web Ads, ...
- **Error metric**: additive-only ⇒ multiplicative-additive
  - Heuristically and rigorously justified
- Pure-DP protocol and a matching lower bound
- Approximate-DP protocol
  - Open directions: A matching lower bound? Or a better algorithm?

Thank you!