Privacy and Security in Distributed Data Markets

Daniel Alabi, Sainyam Galhotra, Shagufta Mehnaz, Zeyu Song, Eugene Wu

SIGMOD 2025 Tutorial

Part 3: Privacy-Preserving Technologies and Security Tools

The Spectrum of Data Marketplace Architectures

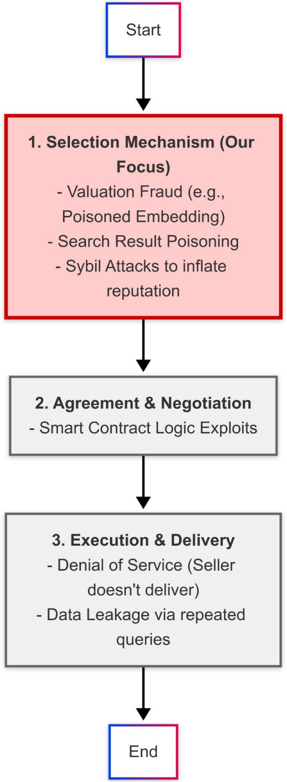
	Centralized Data Storage (Data is Pooled)	Distributed Data Storage (Data Stays Sovereign)
Centralized Governance (Single, Trusted Arbiter)	The Traditional Hub ● Classic data warehouse model ● High trust in one operator required	The Federated Orchestrator ● Data is federated, not pooled ● A central company still manages rules & access
Decentralized Governance (Automated, "Smart" Arbiter)	The Governed Pool ● Data is pooled, but governed by code/community ● A niche but emerging model	⭐ The Sovereign Exchange ● Data Sovereignty by Design ● Transaction Integrity via Arbiter

The Distributed Attack Surface: A Lifecycle View

While a distributed model solves the central honeypot problem, it introduces new, subtle vulnerabilities across the entire transaction lifecycle.

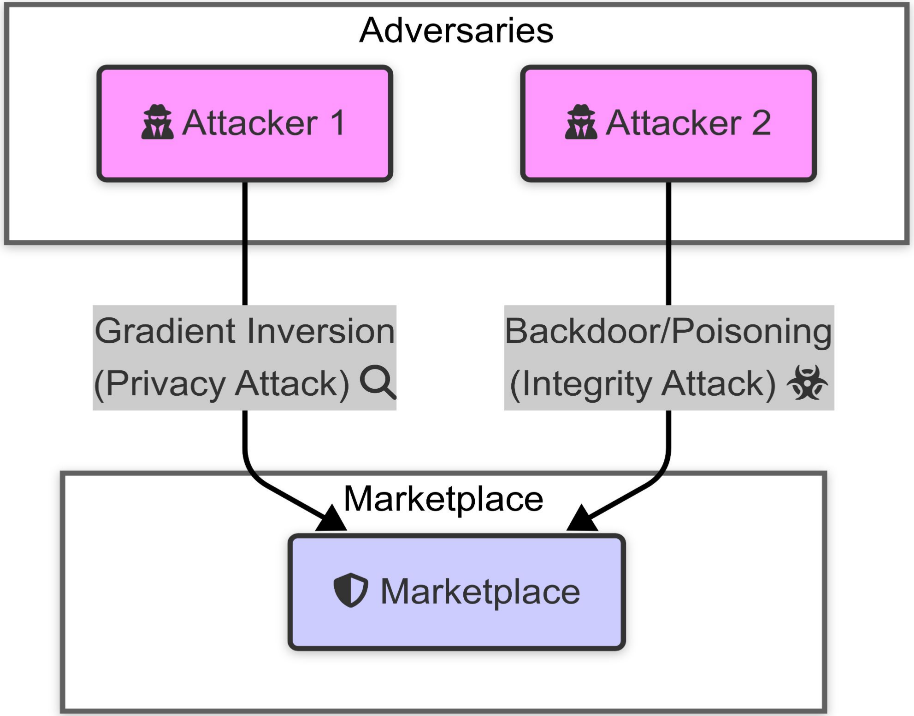
The initial Selection Mechanism is the most critical point of failure. If a buyer is deceived here, the security of the rest of the process is irrelevant.

Therefore, our work focuses on this phase: securing the mechanisms for data Valuation and Retrieval.



The Security Gauntlet: A Marketplace Under Siege

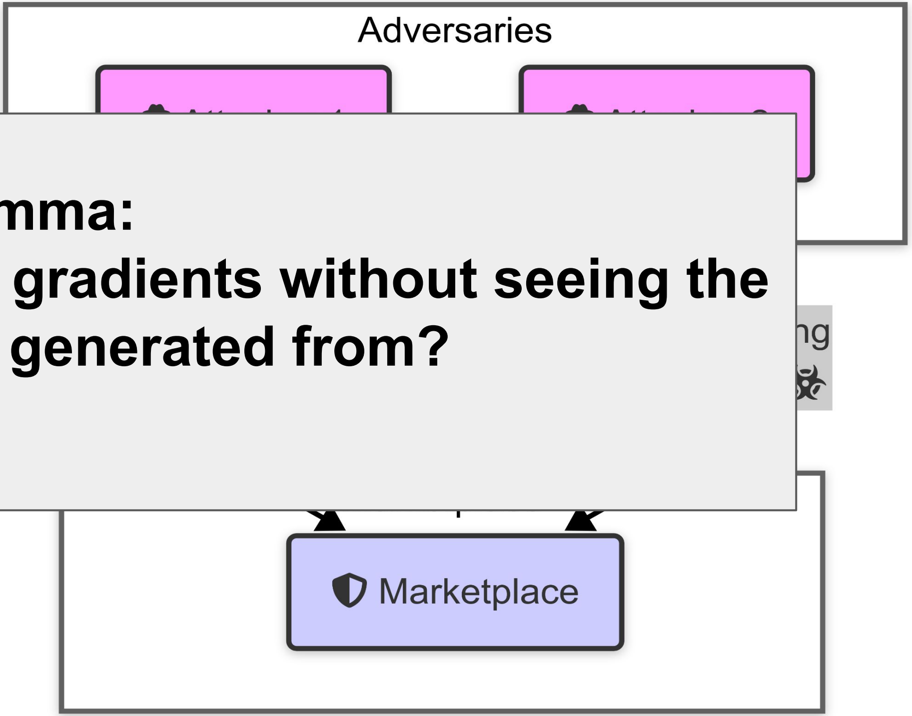
Threat Category	The Adversary's Goal
Privacy Attacks	To reconstruct sensitive, private training data from the shared gradient.
Integrity Attacks	To corrupt the model's performance or install a hidden, malicious trigger.



The Security Gauntlet: A Marketplace Under Sieg

Threat Category	The Adversary's Goal
Privacy	
Integrity Attacks	hidden, malicious trigger.

The Fundamental Dilemma:
How can a Buyer trust gradients without seeing the private data they were generated from?

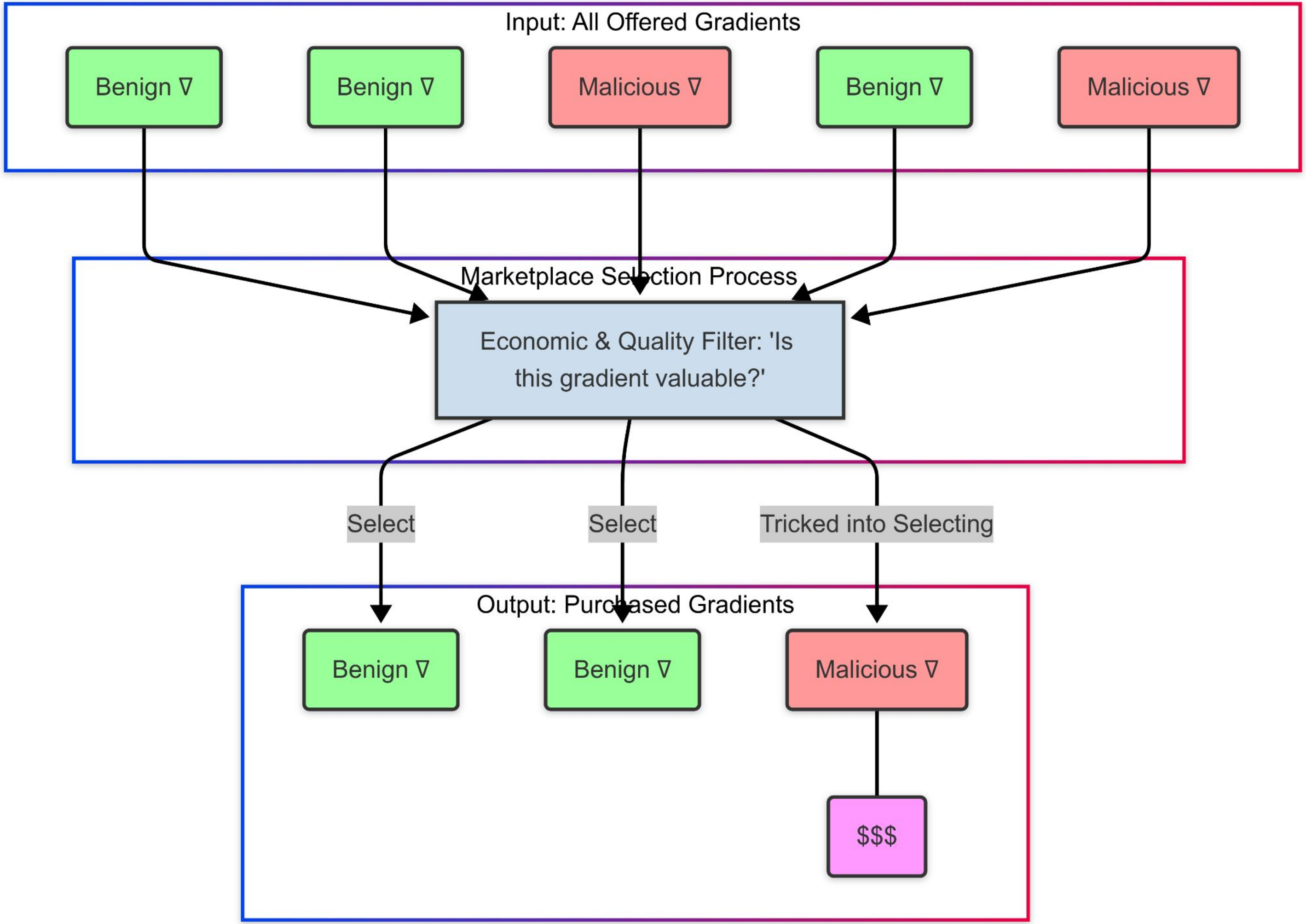


Where Economics Meets Security

In the real world, the marketplace can't just accept everything. Two economic realities force it to be selective:

- High Computational Cost: It's expensive for Sellers to generate gradients.
- Limited Buyer Budget: The Buyer cannot afford to purchase every gradient.

This means every marketplace must have an **Economic and Quality Filter**—a selection mechanism to decide which gradients are worth buying.


Input: All Offered Gradients
Benign ∇
Benign ∇
Malicious ∇
Benign ∇
Malicious ∇
Marketplace Selection Process
Economic & Quality Filter: 'Is this gradient valuable?'
Select
Select
Tricked into Selecting
Output: Purchased Gradients
Benign ∇
Benign ∇
Malicious ∇
$$$

Key Insight: The Filter is the New Battlefield

This selection filter, designed to ensure efficiency and quality, becomes the primary new attack surface.

The sophisticated adversary's goal is no longer just to create a harmful gradient, but to create a harmful gradient that looks beneficial to the filter.

RESEARCH-ARTICLE | OPEN ACCESS

martFL: Enabling Utility-Driven Data Marketplace with a Robust and Verifiable Federated Learning Architecture

Authors: Qi Li, Zhuotao Liu, Qi Li, Ke Xu | Authors Info & Claims

CCS '23: Proceedings of the 2023 ACM SIGSAC Conference on Computer and Communications Security • Pages 1496 - 1510
https://doi.org/10.1145/3576915.3623134

Published: 21 November 2023 Publication History

Check for updates

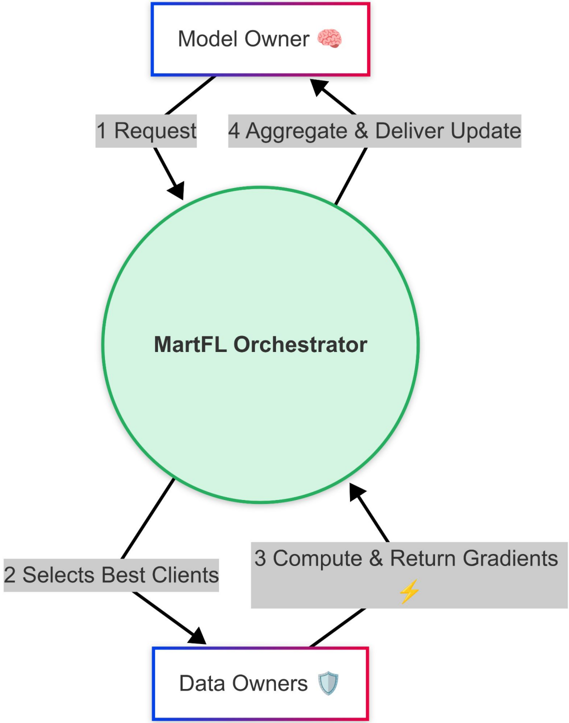
The MartFL Lifecycle: From Request to Improven

Request: A model owner submits a training task to the marketplace.

Select: The MartFL Orchestrator uses its two-phase filter to select the most valuable data owners.

Train: The selected owners compute gradients on their private data.

Improve: MartFL aggregates the gradients, ensures fair payment, and delivers a single, powerful update to the model owner.



The New Threat: Malicious Gradient Attacks

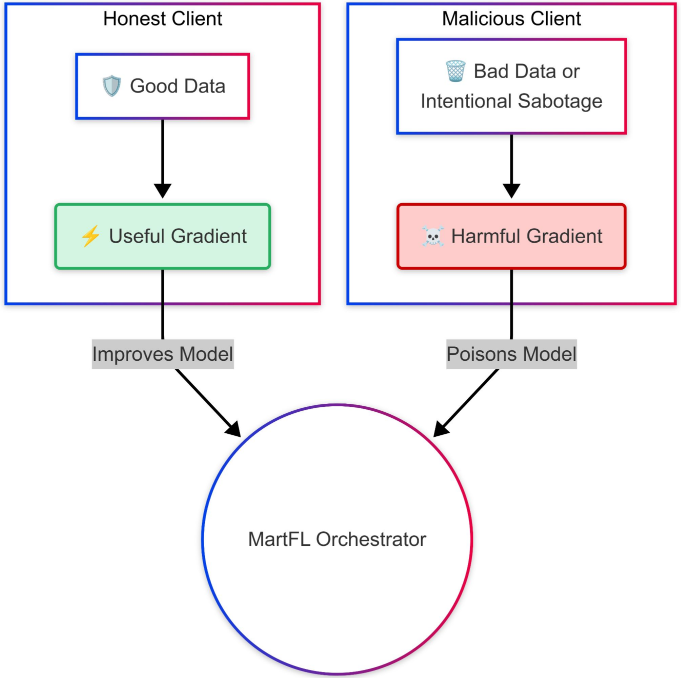
In a gradient marketplace, the threat shifts from faking value to actively sabotaging the training process.

A Malicious Client's Goal:

- Get paid for contributing nothing of value.
- Poison the global model, reducing its accuracy for their own benefit.

The Method:

Submit useless or deliberately harmful gradients instead of honest ones.



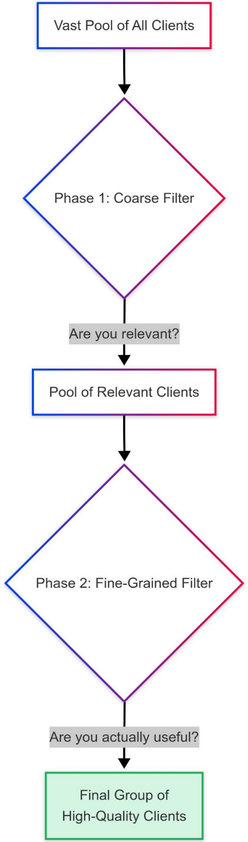
MartFL Defense: The Two-Phase Selection Filter

Coarse-Grained Filter (The Relevance Check):

Uses lightweight data profiles to quickly find clients whose data is topically relevant.

Fine-Grained Filter (The Quality & Behavior Test):

Uses a small "probe model" to test the actual utility of a client's contribution.



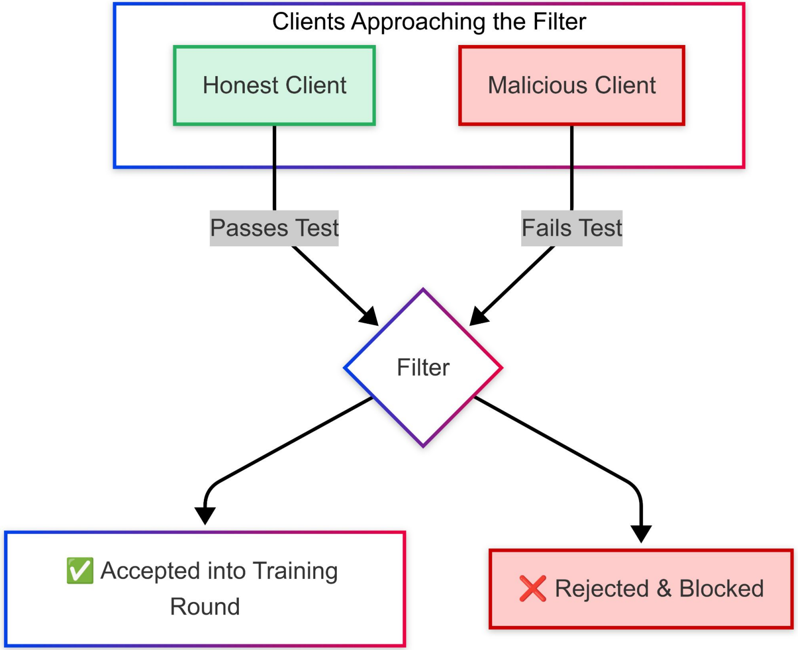
How MartFL Filters Malicious Gradients

The Fine-Grained Filter acts as a behavioral firewall.

It "auditions" each client by sending a small probe task.

An honest client provides a useful gradient, improving the probe model's performance. They pass the test.

A malicious client provides a useless or harmful gradient. The probe model's performance stagnates or drops. They fail the test and are rejected.



The Potential Flaw: Can Similarity Be Fooled?

The MartFL filtering mechanism is clever, but it relies on one critical assumption:

That "low similarity" is a reliable signal for "malicious intent."

The Critical Question:

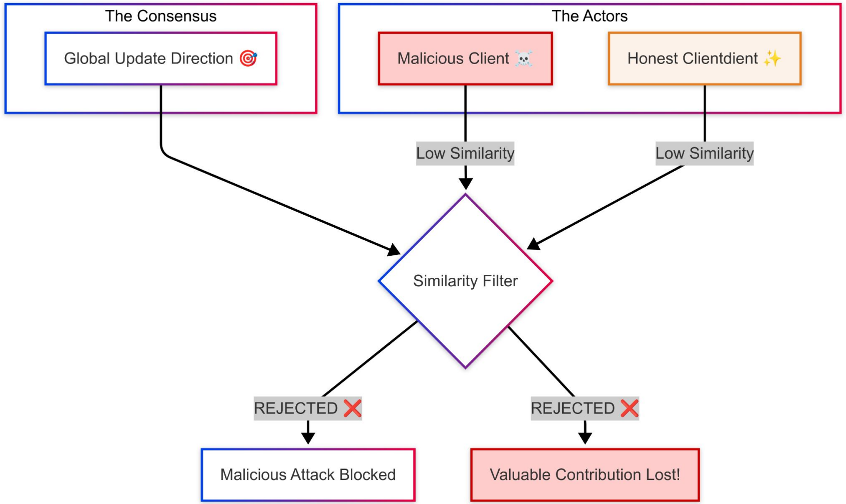
What if a client is honest but their data is unique and valuable? Their gradient might be beneficial but point in a different direction, causing the system to mistakenly reject them.

The Potential Flaw: Can Similarity Be Fooled?

This leads to two key research questions for our analysis:

Robustness: How well does similarity filtering actually detect various malicious attacks?

Fairness: Does this filtering mechanism unfairly penalize honest clients who hold valuable, non-mainstream (outlier) data?



Evaluating the Filter: Marketplace Framework

To understand the true vulnerability, we must go beyond traditional ML security metrics, assessing the filter's impact on the entire marketplace ecosystem, measuring not just security, but economic health and fairness.

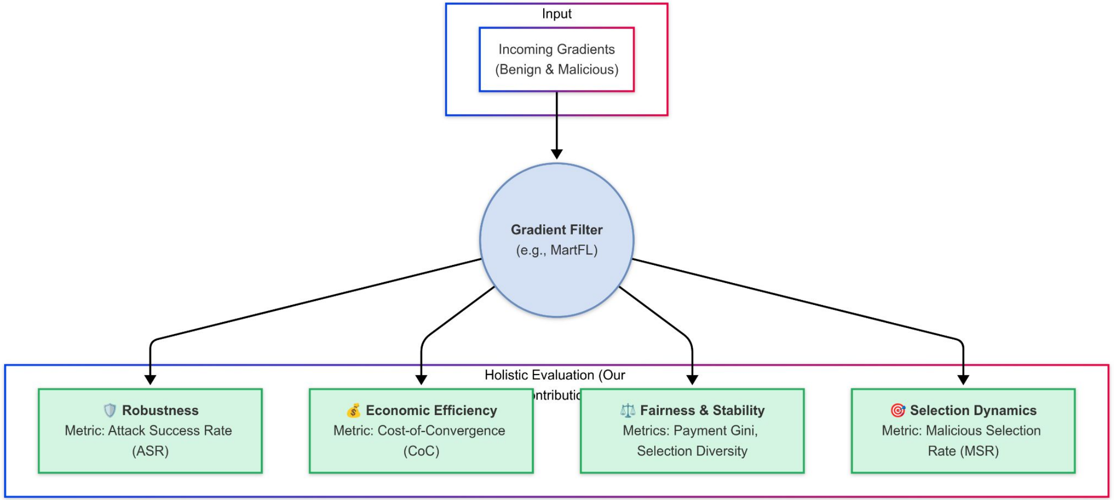
Key Evaluation Dimensions:

Robustness: Does the filter stop the attack? (Traditional Metric)

Economic Efficiency: What is the true cost for the buyer to achieve their goal?

Fairness & Stability: Are honest sellers treated fairly, or are they penalized?

Selection Dynamics: Who is the filter actually selecting, and how often is it fooled?



Test: Can the Marketplace Detect a Backdoor?

We simulated a Backdoor Attack to test the marketplace's security.

The Attack:

Goal: Install a hidden trigger in the model.

Method: Malicious sellers submit gradients from mislabeled, "triggered" data (e.g., cats with a white square are labeled as dogs).

Experimental Setup

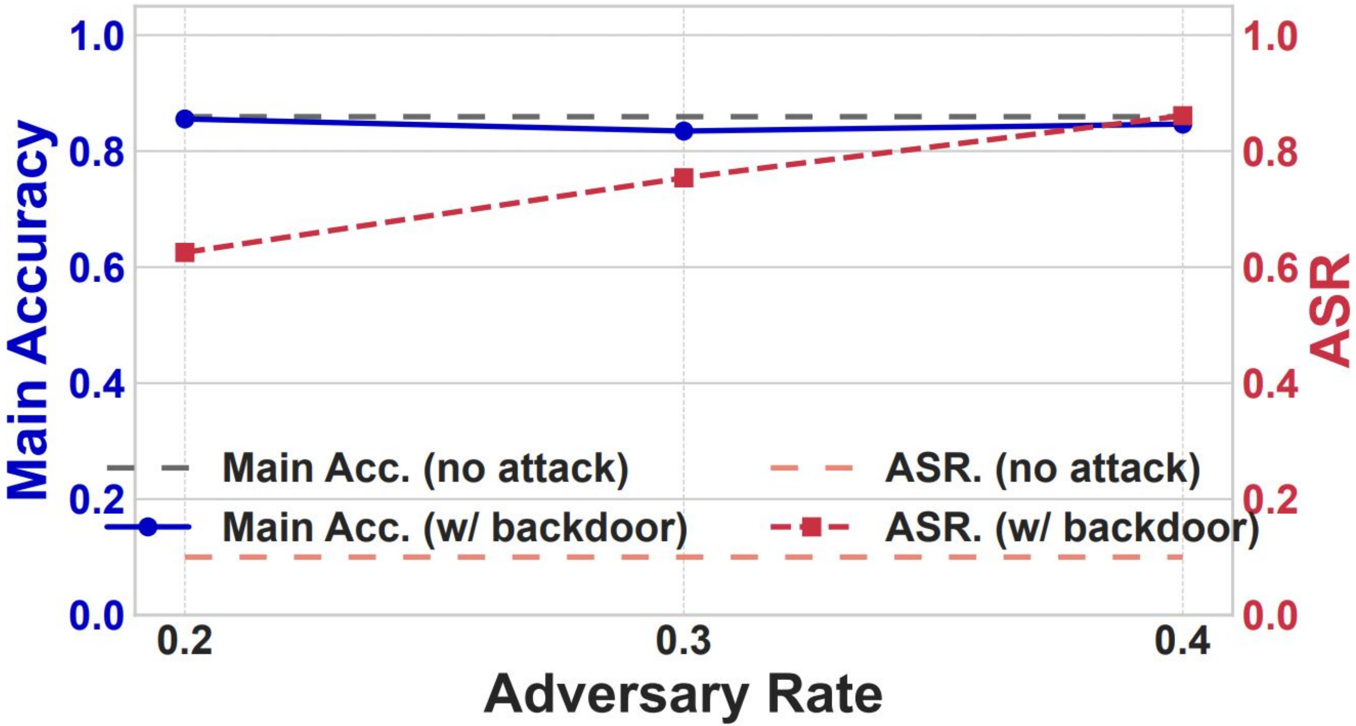
The Adversary: A Stealthy Backdoor Attacker

The Trick: To avoid detection, the attacker doesn't poison all their data. They poison only a small fraction (20%) of their local dataset.

The Effect: This makes their overall gradient more similar to benign gradients, making it harder for a similarity-based filter to spot the manipulation.

The Goal: Pass the filter, get paid, and install a hidden backdoor.

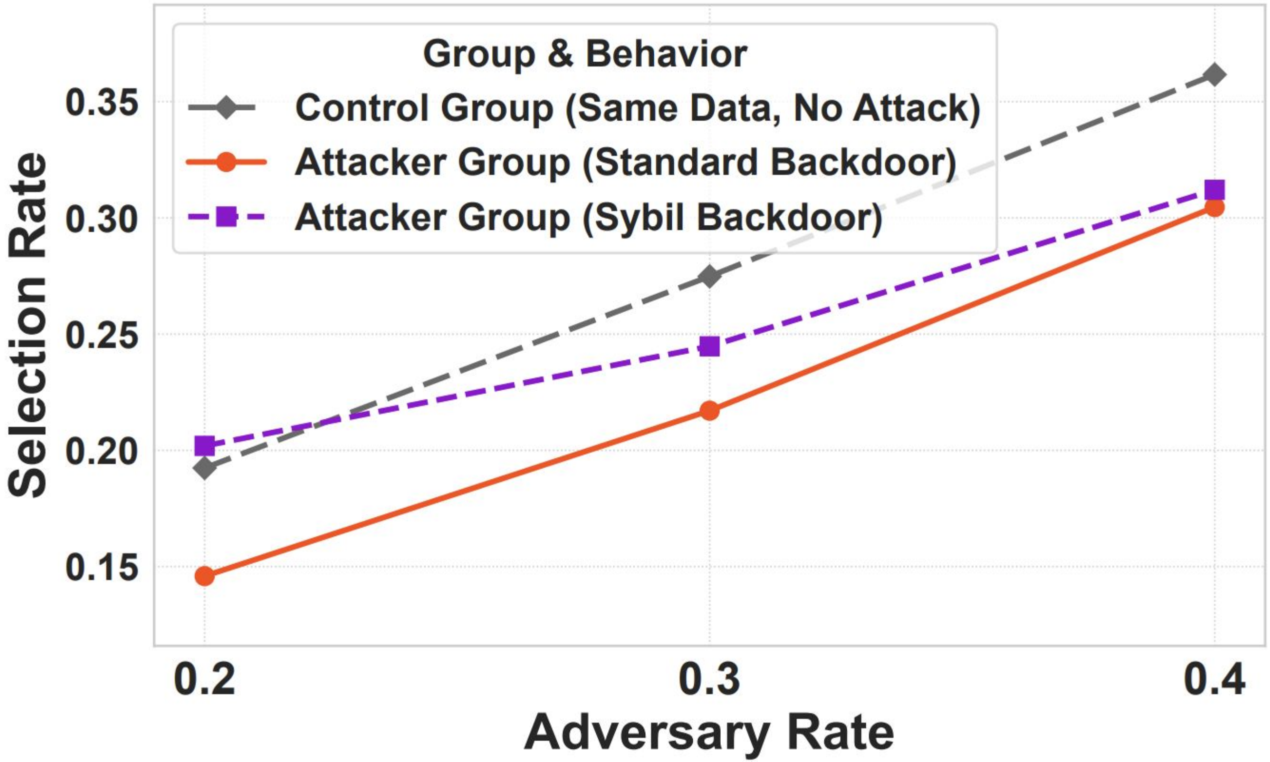
Result



1.0
0.8
0.6
0.4
0.2
0.0
Main Accuracy
1.0
0.8
0.6
0.4
0.2
0.0
ASR
0.2
0.3
0.4
Adversary Rate
Main Acc. (no attack)
Main Acc. (w/ backdoor)
ASR. (no attack)
ASR. (w/ backdoor)

Mechanism of Failure: Why the Filter Was Fooled

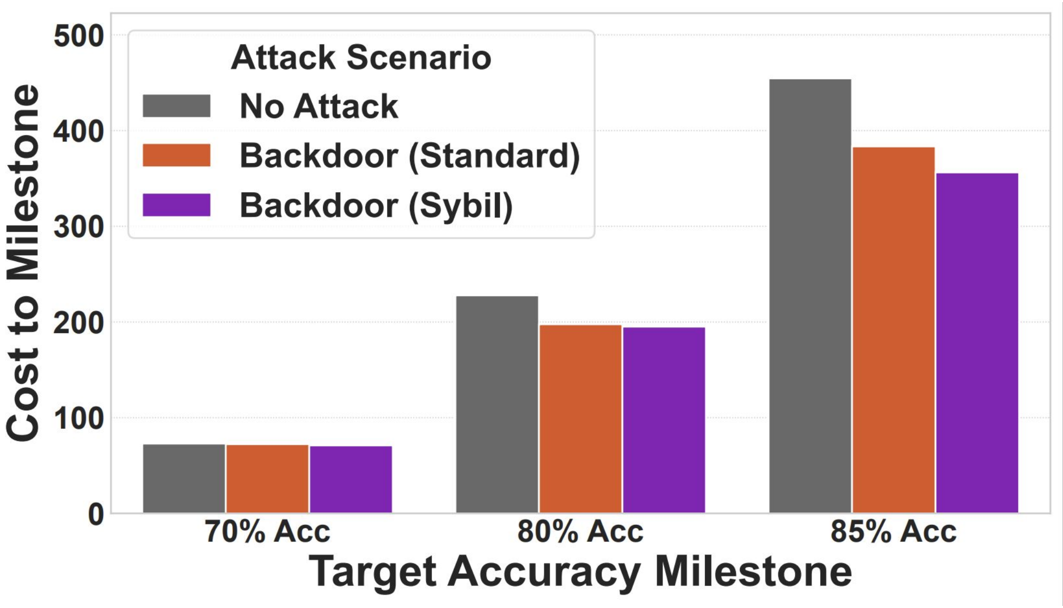
Why Was the Filter Fooled? A Look at Selection Rates.



"Deceptive Efficiency" of an Attacked Market

- The Sybil-attacked market reaches the target accuracy with 23% less cost (fewer gradients purchased).

- From a purely economic standpoint, the attacked market looks more efficient. A buyer optimizing solely for cost would inadvertently prefer the compromised environment. This makes the attack even harder to detect through economic signals.

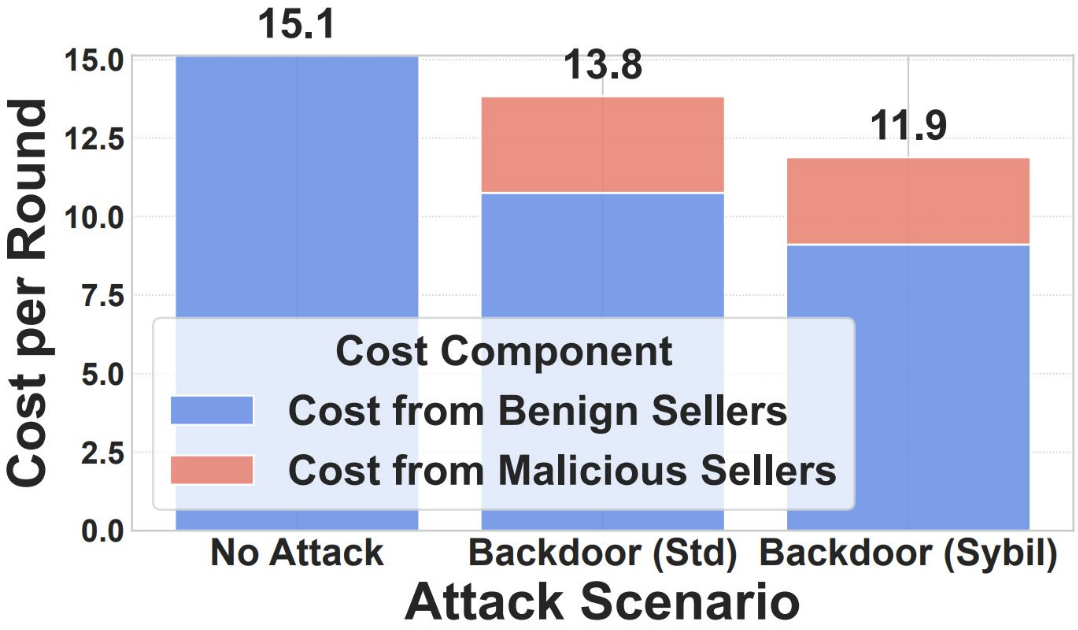


Economic Fallout: Who Really Pays the Price?

No Attack: Honest sellers earn 100% of the revenue.

Sybil Attack: Honest seller revenue plummets by 40%. Attackers successfully siphon off nearly a quarter of all payments

The market isn't more efficient. Attackers are simply crowding out and defunding honest contributors, creating an unsustainable economy.



A Core FL Challenge: Data Heterogeneity

Anyone who has worked with Federated Learning knows the biggest challenge: **Non-IID Data**.

The FL Analogy in a Marketplace:

FL "Client Selection" = Marketplace "Data Discovery"

The Goal is the Same: To select a subset of participants whose data will be most beneficial for the global model's objective.

The Importance of Pre-selection / Discovery:

In both FL and marketplaces, a pre-selection or discovery phase is crucial. The goal is to identify a pool of sellers/clients whose data distribution is most relevant to the task at hand.

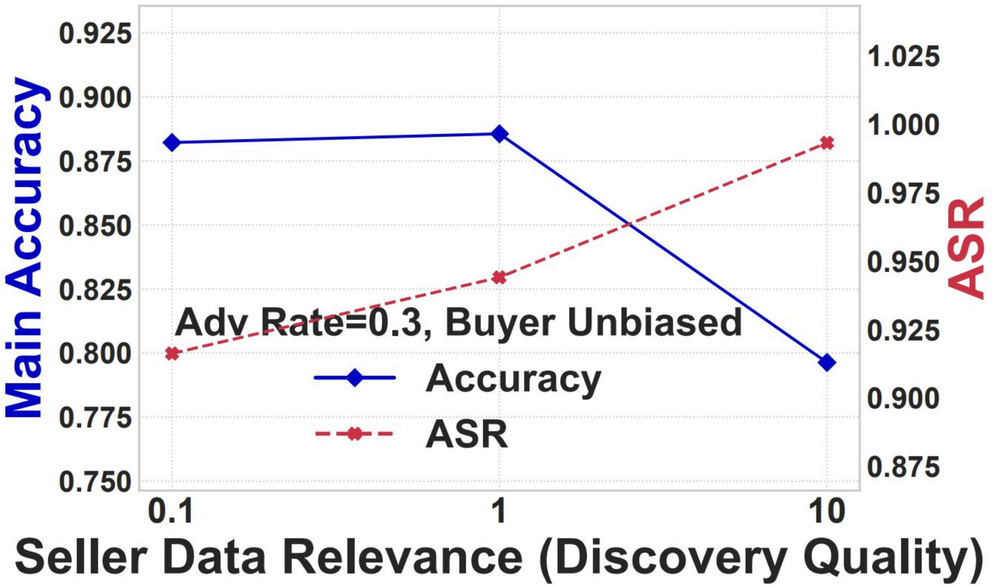
Hypothesis: The more relevant the initial pool of sellers, the better the final model. But how robust is this process to attack?

Data Discovery Dilemma: Diversity vs. Security

Homogenous Data (High Relevance): The similarity filter works reasonably well.

Heterogeneous Data (High Diversity): As the data distribution of the seller pool becomes more diverse, the filter's ability to spot malicious outliers collapses.

The Consequence: The Attack Success Rate (ASR) climbs towards 99% because the filter cannot distinguish "benign heterogeneity" from "malicious intent."



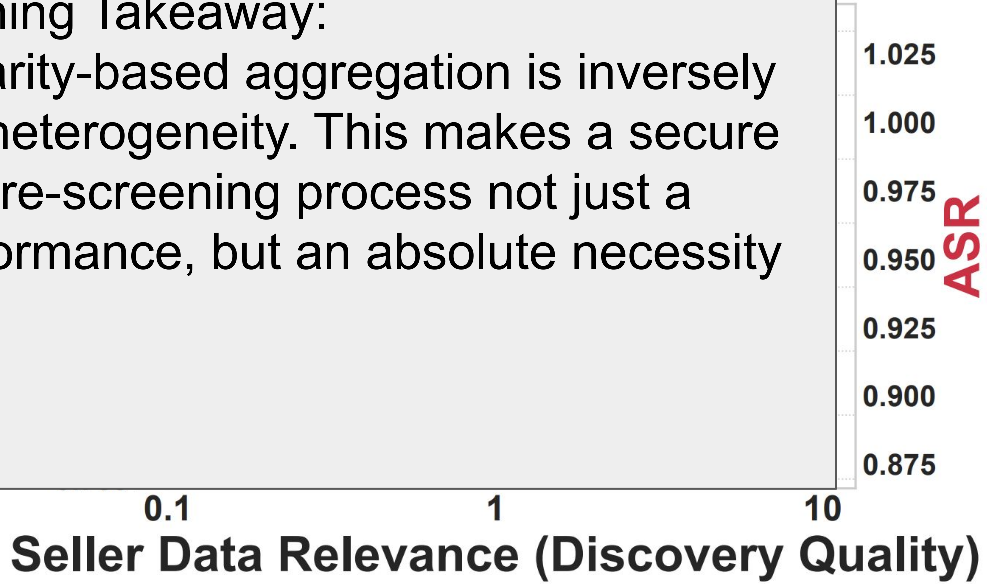
Data Disc rity

Homogenous Data
similarity filter wor

Heterogeneous Da
data distribution of
more diverse, the
malicious outliers

The Consequence
Rate (ASR) climbs
the filter cannot distinguish "benign
heterogeneity" from "malicious intent."

The Federated Learning Takeaway:
The security of similarity-based aggregation is inversely proportional to data heterogeneity. This makes a secure data discovery and pre-screening process not just a nice-to-have for performance, but an absolute necessity for security.



Conclusion & Future Directions

Our investigation into gradient marketplaces reveals critical challenges for building secure, decentralized AI systems.

1. The Attack Surface Has Shifted.

The primary vulnerability is not just the model, but the marketplace's economic and selection mechanisms.

2. Standard Metrics are Deceptive.

High model accuracy and low cost can mask catastrophic security failures and unfair economic outcomes.

3. Similarity-Based Defenses are Not a Silver Bullet.

They are fundamentally vulnerable to mimicry attacks and struggle most in the realistic, heterogeneous environments they are designed for.

Path Forward: Building on Robust Gradient

To build truly secure and equitable marketplaces, future work must move beyond simple similarity checks. We need to focus on:

- **Orthogonal Trust Signals**: Integrating seller reputation, transaction history, and data provenance to make more holistic trust decisions.
- **Multi-Stage Filtering**: Designing a defense-in-depth pipeline that combines anomaly detection, similarity checks, and impact analysis.
- **Incentive-Compatible Mechanisms**: Creating reward and selection systems that are provably resilient to strategic manipulation and fairly compensate true value.

Differentially Private Task-based Search

Kitana's Basic Algorithm

```
D = initial training dataset
for next augmentation α
    if eval(apply A to D) is best so far
        Keep α in A
return best A
```

Greedy

Use sketches

But can we enforce differential privacy?

Differential Privacy

Privatization: Differential Privacy(DP) Algorithm

$$\Pr[M(D)\in S] \leq \exp(\boxed{\epsilon}) * \Pr[M(D')\in S] \boxed{+}\, \delta$$

Informally, an algorithm satisfies DP if no single record can be inferred

- Hides individuals in a dataset by adding noise to results
- Each query consumes part of a dataset’s finite budget
- Consumed budget ∝ noise added to result

Differential Privacy: Privacy Budget

Privacy budget: (ϵ, δ)

SELECT SUM(Y) FROM D

D

A	Y
a_1	1
a_1	2

Remaining budget:

(ϵ, δ)

Differential Privacy: Privacy Budget

Privacy budget: (ϵ, δ)

SELECT SUM(Y) FROM D

$1 + 2 + 1 + \textit{noise}_{\epsilon,\,\delta} = \boxed{4.2}$

D

A	Y
a_1	1
a_1	2

Remaining budget:

(0, 0)

Differential Privacy: Privacy Budget

Privacy budget: (ϵ, δ)

SELECT SUM(Y*Y) FROM D

D

A	Y
a_1	1
a_1	2

Remaining budget:

(0, 0)

Differential Privacy: Privacy Budget

Privacy budget: (ϵ, δ)

SELECT SUM(Y*Y) FROM D

No privacy budget, cannot access

D

A	**Y**
a_1	1
a_1	2

Remaining budget:

(0, 0)

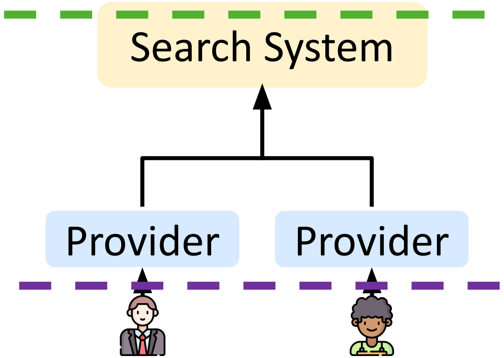
Differential Privacy Mechanisms Available

Uses budget on every model eval

GDP

Too much noise to be useful

LDP



Data Task

ML data augmentation search

- More samples to union with.
- More features to join with.

Example: Predicting churn

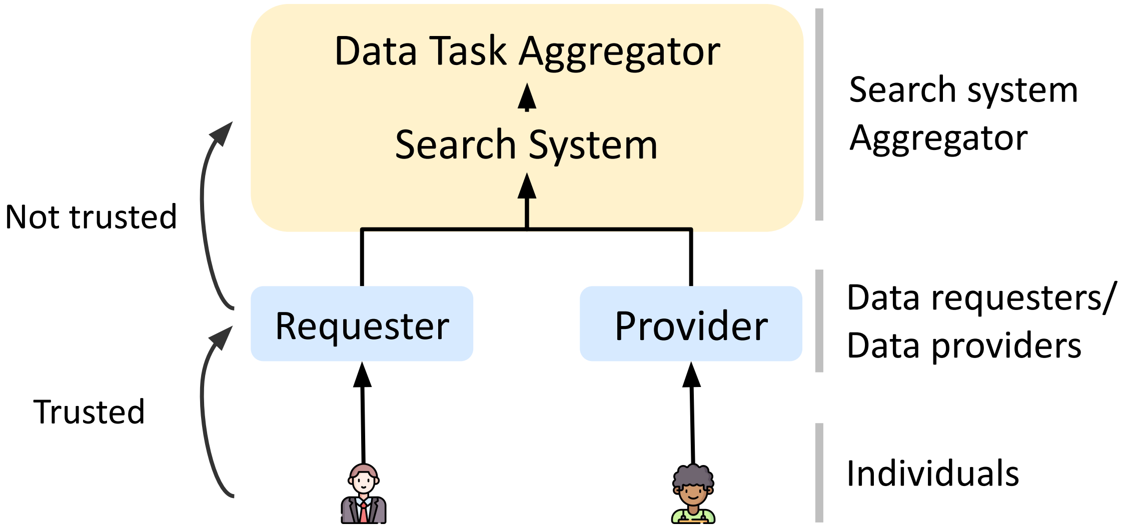
Churned	Customer	Subscription Date	Most Visited	Unemployment Rate
Yes	Alice	Jan 2023	Products	6.5%
No	Bob	May 2023	Support	3.2%
Yes	Charlie	Feb 2023	Support	8.1%
No	David	Jan 2023	Home	6.5%

DP ML Data Augmentation: Input and Output

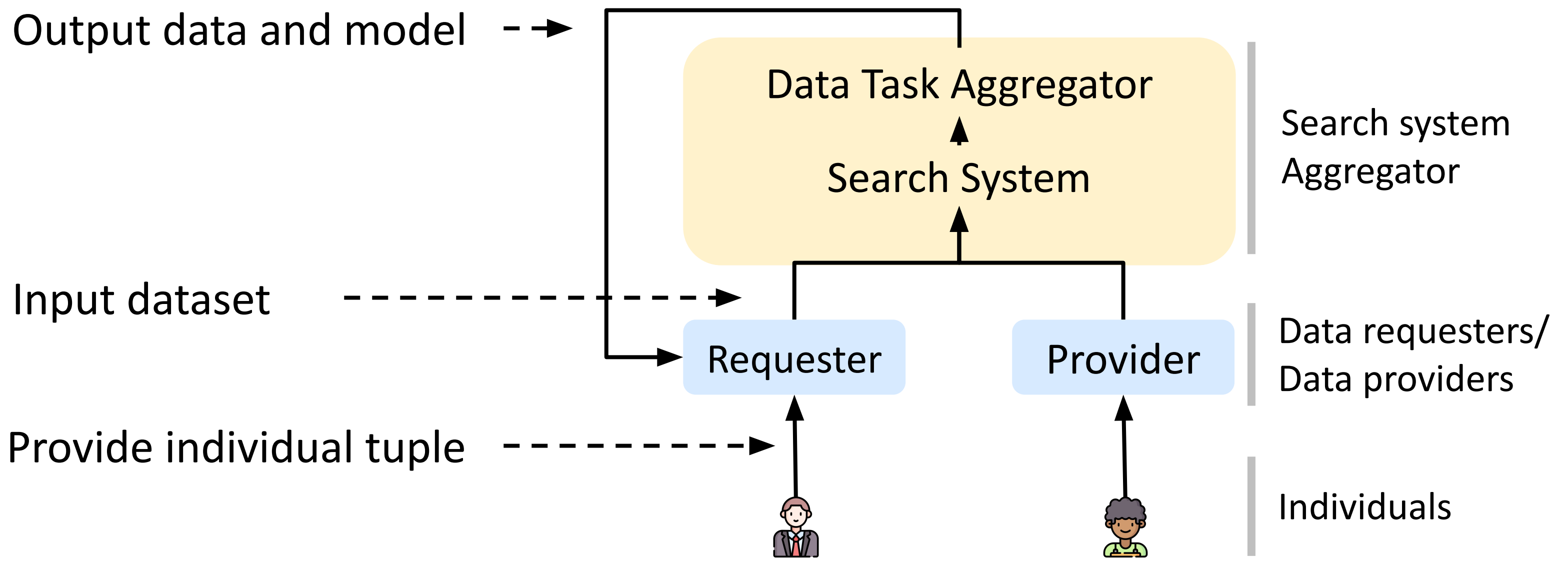
Want to find health data to improve cardiac prediction models

Patients don’t trust Google to use health data for ads.

Patients trust their health tracking App, like Fitbit.



DP ML Data Augmentation: Input and Output

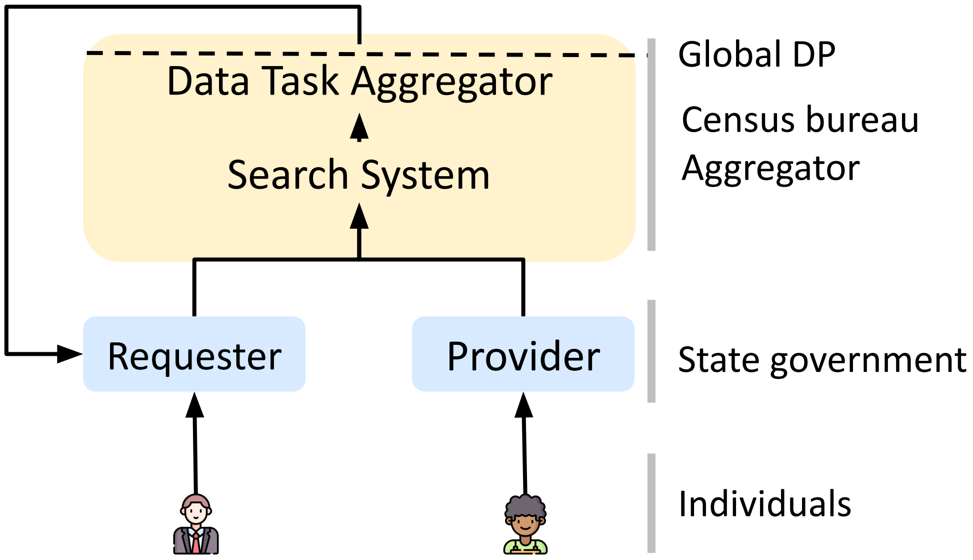


Existing Approach Limitations: Global DP

Global DP mechanisms add noise before releasing the output.

Evaluating each combination drains privacy budget.

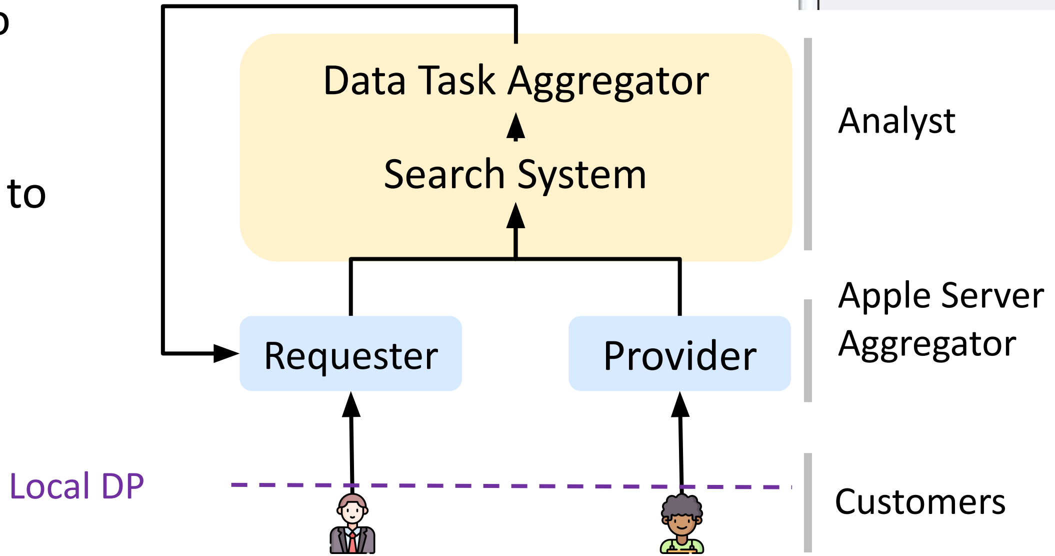
Exponential combinations of join/union-compatible sets.



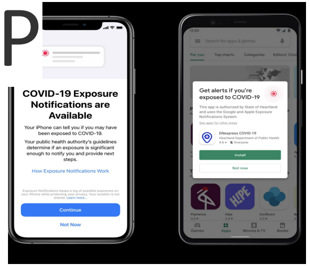
Existing Approach Limitations: Local DP

Local DP mechanisms add noise to each customer's data.

Augmentations too noisy, difficult to distinguish useful ones.

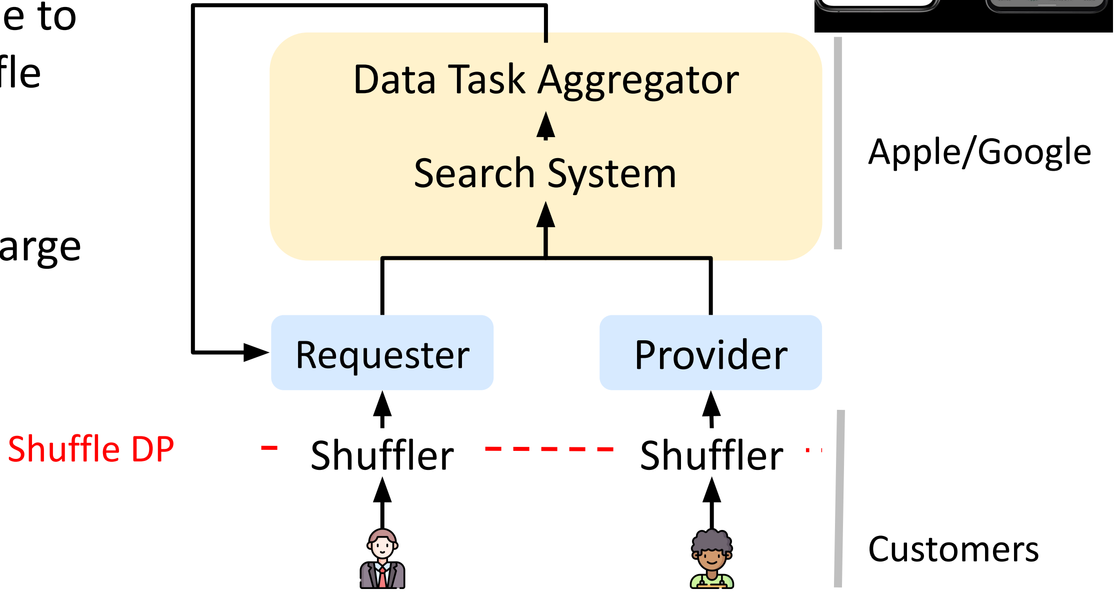


Existing Approach Limitations: Shuffle DP

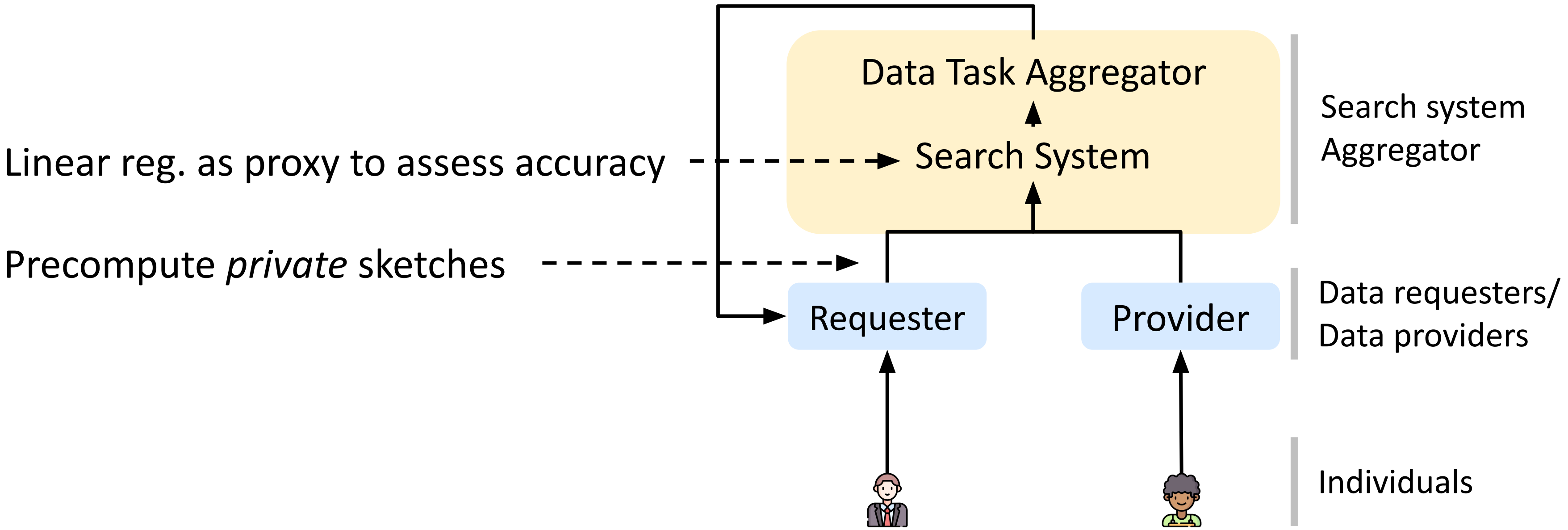


Shuffle DP mechanisms add noise to each customer's data, then shuffle to enhance privacy.

Only enhance privacy levels for large datasets.



Sketch-based Approach



Sketch-based Search

Linear regression has closed form solution $\hat{\beta} = \boxed{(X^T X)}^{-1} \boxed{X^T y}$

$$X^T X = \begin{bmatrix} \sum x_1 x_1 & \dots & \sum x_1 x_m \\ \dots & \dots & \dots \\ \sum x_m x_1 & & \sum x_m x_m \end{bmatrix}$$

Sketch-based Search

How to compute sum of pairwise product between features?

D

A	Y
a_1	1
a_1	2

Compute aggregates as sketches

Monomial semi-ring

A	sum(Y^2)	sum(Y)	count
a_1	1^2+2^2	1+2	2

Sum of 0^{th}, 1^{st}, 2^{nd}-order monomials

Sketch-based Search

Linear regression on D⋈R requires computing $\boxed{\sum 1, \sum B, \sum Y, \sum BY}$

D

A	Y
a_1	1
a_1	2

⋈

R

A	B
a_1	1
a_1	2

D ⋈ R

A	Y	B
a_1	1	1
a_1	1	2
a_1	2	1
a_1	2	2

= 9

Sketch-based Search

Linear regression on D⋈R requires computing $\sum 1, \sum B, \sum Y, \boxed{\sum BY}$

D

A	Y
a_1	1
a_1	2

⋈

R

A	B
a_1	1
a_1	2

D ⋈ R

A	Y	B
a_1	1	1
a_1	1	2
a_1	2	1
a_1	2	2

= 9

Sketch-based Search

Linear regression on D⋈R requires computing $\sum 1, \sum B, \sum Y, \boxed{\sum BY}$

D

A	Y
a_1	1
a_1	2

1^{st}-order

A	sum(Y)
a_1	1+2

⋈

R

A	B
a_1	1
a_1	2

D ⋈ R

A	Y	B
a_1	1	1
a_1	1	2
a_1	2	1
a_1	2	2

= 9

Sketch-based Search

Linear regression on D⋈R requires computing $\sum 1$, $\sum B$, $\sum Y$, $\boxed{\sum BY}$

D

A	Y
a_1	1
a_1	2

1^{st}-order

A	sum(Y)
a_1	1+2

⋈

1^{st}-order

A	sum(B)
a_1	1+2

R

A	B
a_1	1
a_1	2

D ⋈ R

A	Y	B
a_1	1	1
a_1	1	2
a_1	2	1
a_1	2	2

= 9

Sketch-based Search

Linear regression on D⋈R requires computing $\sum 1$, $\sum B$, $\sum Y$, $\boxed{\sum BY}$

D

A	Y
a_1	1
a_1	2

1st-order

A	sum(Y)
a_1	1+2

$\otimes$

1st-order

A	sum(B)
a_1	1+2

R

A	B
a_1	1
a_1	2

2nd-order

A	sum(B×Y)
a_1	3×3

= 9

Sketch-based Search

Linear regression on D⋈R requires computing $\sum 1$, $\sum B$, $\sum Y$, $\boxed{\sum BY}$

D

A	Y
a_1	1
a_1	2

1st-order

A	sum(Y)
a_1	1+2

$\otimes$

1st-order

A	sum(B)
a_1	1+2

R

A	B
a_1	1
a_1	2

D ⋈ R

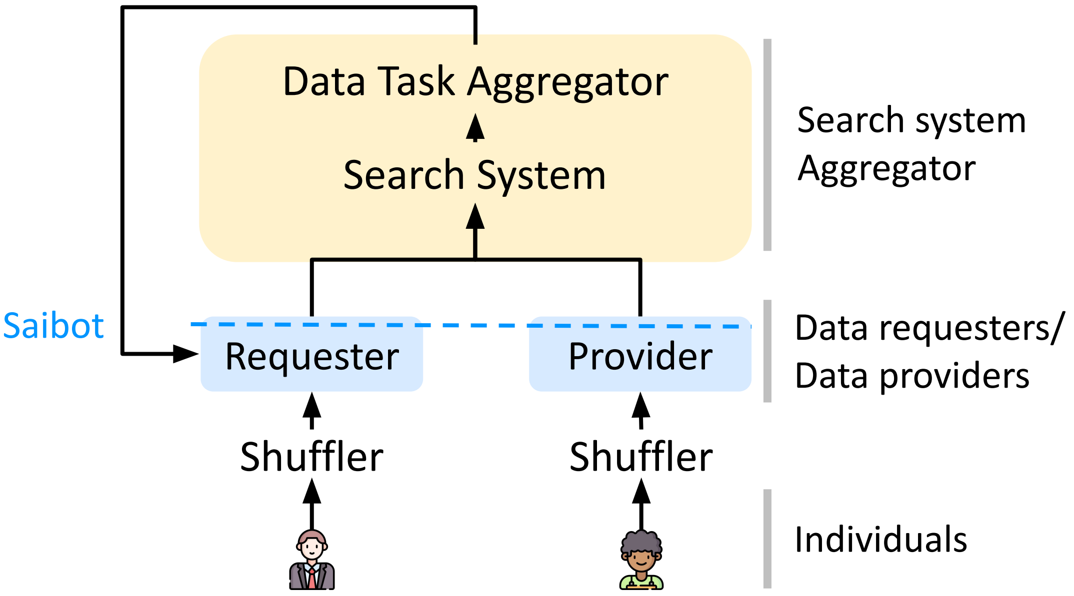
A	Y	B
a_1	1	1
a_1	1	2
a_1	2	1
a_1	2	2

= 9

Saibot: Our Contribution

Factorized Privacy Mechanism

Privatize sketches so downstream search will be privatized.



Intuition: aggregate datasets as much as possible before adding noise to them.

Saibot: Technical Details

- Factorized Privacy Mechanism (FPM).
- Noise allocation optimization.
- Unbiased estimation.
- Proofs

Saibot: Technical Details

- Factorized Privacy Mechanism (FPM).
- Noise allocation optimization.
- Unbiased estimation.
- Proofs

Saibot: Assumptions

The schema and join keys for datasets owned by providers are public

- ❖ Oblivious intersection techniques can be applied.

All tuples are L2 bounded by B (for analysis)

- ❖ Categorical features numericalized

FPM:Privatize sketches with privacy budget (ϵ, δ)

Use existing DP query engine

D

A	**Y**
a_1	1

$\xrightarrow{\textbf{Q}}$

A	**sum(Y^2)**	**sum(Y)**	**count(Y)**
a_1	1^2	1	1

Q: SELECT SUM(**Y**^{2}), SUM(**Y**), COUNT(**Y**) from **D** GROUP BY **A**

FPM:Privatize sketches with privacy budget (ϵ, δ)

Use existing DP query engine

D

A	Y
a_1	1

Q

A	sum(Y^2)	sum(Y)	count(Y)
a_1	1^2	1	1

Neighbour

D'

A	Y
a_1	2

Q

A	sum(Y^2)	sum(Y)	count(Y)
a_1	2^2	2	1

Q: SELECT SUM(**Y**^{2}), SUM(**Y**), COUNT(**Y**) from **D** GROUP BY **A**

FPM:Privatize sketches with privacy budget (ϵ, δ)

Use existing DP query engine

D

A	Y
a_1	1

$\xrightarrow{\mathbf{Q}}$

A	sum(Y^2)	sum(Y)	count(Y)
a_1	1^2	1	1

Neighbour

L2 distance

D’

A	Y
a_1	2

$\xrightarrow{\mathbf{Q}}$

A	sum(Y^2)	sum(Y)	count(Y)
a_1	2^2	2	1

Sensitivity of Q: $\Delta(\mathbf{Q}) = \|\mathbf{Q}(\mathbf{D}) - \mathbf{Q}(\mathbf{D'})\|_2$

FPM:Privatize sketches with privacy budget (ϵ, δ)

Use existing DP query engine

D

A	**Y**
a_1	1

$\xrightarrow{\textbf{Q}}$

A	**sum(Y²)**	**sum(Y)**	**count(Y)**
a_1	1^2	1	1

$$\mathcal{N}\left(0, \frac{\sqrt{2\ln(1.25/\delta)}\Delta(Q)}{\boxed{\epsilon}}\right)$$

Budget

Q: SELECT SUM(**Y**^{2}), SUM(**Y**), COUNT(**Y**) from **D** GROUP BY **A**

FPM:Privatize sketches with privacy budget (ϵ, δ)

Use existing DP query engine

D

A	**Y**
a_1	1

$\xrightarrow{\mathbf{Q}}$

A	**sum(Y²)**	**sum(Y)**	**count(Y)**
a_1	1^2+e_1	$1+e_2$	$1+e_3$

e_1, e_2, e_3 $\quad \mathcal{N}\left(0, \frac{\sqrt{2\ln(1.25/\delta)}\Delta(Q)}{\epsilon}\right)$

Q: SELECT SUM(**Y**^{2}), SUM(**Y**), COUNT(**Y**) from **D** GROUP BY **A**

Naive Solution Limitation:Combining Sketches

A	sum(C^2)	sum(C)	count(C)
a_1	$2^2+e'_1$	$2+e'_2$	$1+e'_3$

A	sum(Y^2)	sum(Y)	count(Y)
a_1	1^2+e_1	$1+e_2$	$1+e_3$

A	sum(C^2)	sum(Y^2)	sum(C×Y)	sum(C)	sum(Y)	count
a_1	$(2^2+e'_1)(1+e_3)$	$(1+e'_3)(1^2+e_1)$	$(2+e'_2)(1+e_2)$	$(2+e'_2)(1+e_3)$	$(1+e'_3)(1+e_2)$	$(1+e'_3)(1+e_3)$

Naive Solution Limitation:Combining Sketches

A	sum(C²)	sum(C)	count(C)
a_1	$2^2+e'_1$	$2+e'_2$	$1+e'_3$

A	sum(Y²)	sum(Y)	count(Y)
a_1	1^2+e_1	$1+e_2$	$1+\mathbf{e_3}$

A	sum(C²)	sum(Y²)	sum(C×Y)	sum(C)	sum(Y)	count
a_1	$(2^2+e'_1)(1+\mathbf{e_3})$	$(1+e'_3)(1^2+e_1)$	$(2+e'_2)(1+e_2)$	$(2+e'_2)(1+\mathbf{e_3})$	$(1+e'_3)(1+e_2)$	$(1+e'_3)(1+\mathbf{e_3})$

Naive Solution Limitation: Combining Sketches

A	sum(C²)	sum(C)	count(C)
a_1	$2^2+e'_1$	$2+e'_2$	$1+e'_3$

A	sum(Y²)	sum(Y)	count(Y)
a_1	1^2+e_1	$1+\mathbf{e_2}$	$1+e_3$

A	sum(C²)	sum(Y²)	sum(C×Y)	sum(C)	sum(Y)	count
a_1	$(2^2+e'_1)(1+e_3)$	$(1+e'_3)(1^2+e_1)$	$(2+e'_2)(1+\mathbf{e_2})$	$(2+e'_2)(1+e_3)$	$(1+e'_3)(1+\mathbf{e_2})$	$(1+e'_3)(1+e_3)$

Naive Solution Limitation:Combining Sketches

A	sum(C²)	sum(C)	count(C)
a_1	$2^2+e'_1$	$2+e'_2$	$1+e'_3$

A	sum(Y²)	sum(Y)	count(Y)
a_1	$1^2+\mathbf{e_1}$	$1+e_2$	$1+e_3$

A	sum(C²)	sum(Y²)	sum(C×Y)	sum(C)	sum(Y)	count
a_1	$(2^2+e'_1)(1+e_3)$	$(1+e'_3)(1^2+\mathbf{e_1})$	$(2+e'_2)(1+e_2)$	$(2+e'_2)(1+e_3)$	$(1+e'_3)(1+e_2)$	$(1+e'_3)(1+e_3)$

Noise Allocation

How to draw noise from different distributions to aggregations?

$sum(Y^2)$	$sum(Y)$	$count(Y)$
1^2+e_1	$1+e_2$	$1+e_3$

Noise Allocation

How to draw noise from different distributions to aggregations?

sum(Y^2)	**sum(Y)**	**count(Y)**
1^2+e_1	$1+e_2$	$1+e_3$

Sensitivity $O(B^2)$	**Sensitivity $O(B)$**	**Sensitivity $O(1)$**

Noise Allocation

How to draw noise from different distributions to aggregations?

sum(Y²)	sum(Y)	count(Y)
1^2+e_1	$1+e_2$	$1+e_3$

$e_1 \sim \mathcal{N}\left(0, \frac{\sqrt{2\ln(1.25/\delta)}B^2}{\epsilon_1}\right)$

$e_2 \sim \mathcal{N}\left(0, \frac{\sqrt{2\ln(1.25/\delta)}B}{\epsilon_2}\right)$

$e_3 \sim \mathcal{N}\left(0, \frac{\sqrt{2\ln(1.25/\delta)}}{\epsilon_3}\right)$

Noise Allocation

How to draw noise from different distributions to aggregations?

sum(Y²)	sum(Y)	count(Y)
1^2+e_1	$1+e_2$	$1+e_3$

$$e_1 \sim \mathcal{N}\left(0, \frac{\sqrt{2\ln(1.25/\delta)}B^2}{\epsilon/3}\right)$$

$$e_2 \sim \mathcal{N}\left(0, \frac{\sqrt{2\ln(1.25/\delta)}B}{\epsilon/3}\right)$$

$$e_3 \sim \mathcal{N}\left(0, \frac{\sqrt{2\ln(1.25/\delta)}}{\epsilon/3}\right)$$

Noise Allocation: Analysis

Bounding linear regression estimator: $|\hat{\beta}_{\boldsymbol{x}} - \tilde{\beta}_{\boldsymbol{x}}| \leq \tau_2 + \frac{\tau_1}{1-\tau_1}\left(\hat{\beta}_{\boldsymbol{x}} + \tau_2\right)$

Noise Allocation: Analysis

Bounding linear regression estimator: $|\hat{\beta}_x - \tilde{\beta}_x| \leq \tau_2 + \frac{\tau_1}{1-\tau_1}\left(\hat{\beta}_x + \tau_2\right)$

Naive Method: $\tau_1 = O\left(\frac{B^4\sqrt{d}\ln(1/\delta)\ln(1/p)}{\epsilon^2 n\widehat{\sigma_{\mathrm{x}}^2}}\right)$ $\tau_2 = O\left(\frac{B^4\ln(1/p)\ln(1/\delta)\sqrt{d\ln(d/p)}}{\epsilon^2\sqrt{n}\widehat{\sigma_{\mathrm{x}}^2}}\right)$

Noise Allocation: Analysis

Bounding linear regression estimator: $|\hat{\beta}_x - \tilde{\beta}_x| \leq \tau_2 + \frac{\tau_1}{1-\tau_1}\left(\hat{\beta}_x + \tau_2\right)$

Naive Method: $\tau_1 = O\left(\frac{B^4\sqrt{d}\ln(1/\delta)\ln(1/p)}{\epsilon^2 n\widehat{\sigma_{\mathrm{x}}^2}}\right)$ $\tau_2 = O\left(\frac{B^4\ln(1/p)\ln(1/\delta)\sqrt{d\ln(d/p)}}{\epsilon^2\sqrt{n}\widehat{\sigma_{\mathrm{x}}^2}}\right)$

Optimization: $\tau_1 = O\left(\frac{B^2\sqrt{d}\ln(1/\delta)\ln(1/p)}{\epsilon^2 n\widehat{\sigma_{\mathrm{x}}^2}}\right), \tau_2 = O\left(\frac{B^2\ln(1/p)\ln(1/\delta)\sqrt{d\ln(d/p)}}{\epsilon^2\sqrt{n}\widehat{\sigma_{\mathrm{x}}^2}}\right)$

Noise Allocation: Analysis

Bounding linear regression estimator: $|\hat{\beta}_x - \tilde{\beta}_x| \leq \tau_2 + \frac{\tau_1}{1-\tau_1}\left(\hat{\beta}_x + \tau_2\right)$

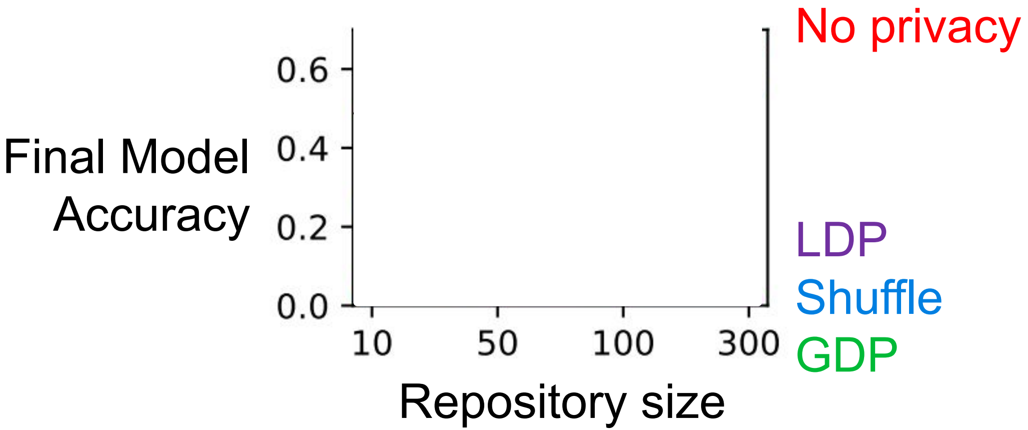
Naive Method: $\tau_1 = O\left(\frac{\boxed{B^4}\sqrt{d}\ln(1/\delta)\ln(1/p)}{\epsilon^2 n\widehat{\sigma_{\mathrm{x}}^2}}\right)$ $\tau_2 = O\left(\frac{\boxed{B^4}\ln(1/p)\ln(1/\delta)\sqrt{d\ln(d/p)}}{\epsilon^2\sqrt{n}\widehat{\sigma_{\mathrm{x}}^2}}\right)$

Optimization: $\tau_1 = O\left(\frac{\boxed{B^2}\sqrt{d}\ln(1/\delta)\ln(1/p)}{\epsilon^2 n\widehat{\sigma_{\mathrm{x}}^2}}\right), \tau_2 = O\left(\frac{\boxed{B^2}\ln(1/p)\ln(1/\delta)\sqrt{d\ln(d/p)}}{\epsilon^2\sqrt{n}\widehat{\sigma_{\mathrm{x}}^2}}\right)$

Reduce the bound on linear regression parameter by O(**B²**)

Prior Mechanisms Don't Scale

To repository size & number of requests
Vary between 10 - 329 NYC Open Datasets in Repo



Prior Mechanisms Don't Scale

To repository size & number of requests
Vary between 10 - 329 NYC Open Datasets in Repo