Differentially Private Algorithms with Correlated data

Amartya Sanyal

Tenure Track Assistant Professor in Machine Learning, University of Copenhagen

Adjunct Assistant Professor, IIT Kanpur

Violation of Privacy in Machine Learning

Violation of Privacy in Machine Learning

- Large machine learning (ML) models often memorise training data

Violation of Privacy in Machine Learning

- Large machine learning (ML) models often memorise training data
- Sophisticated attacks can leak these private and sensitive data

Violation of Privacy in Machine Learning

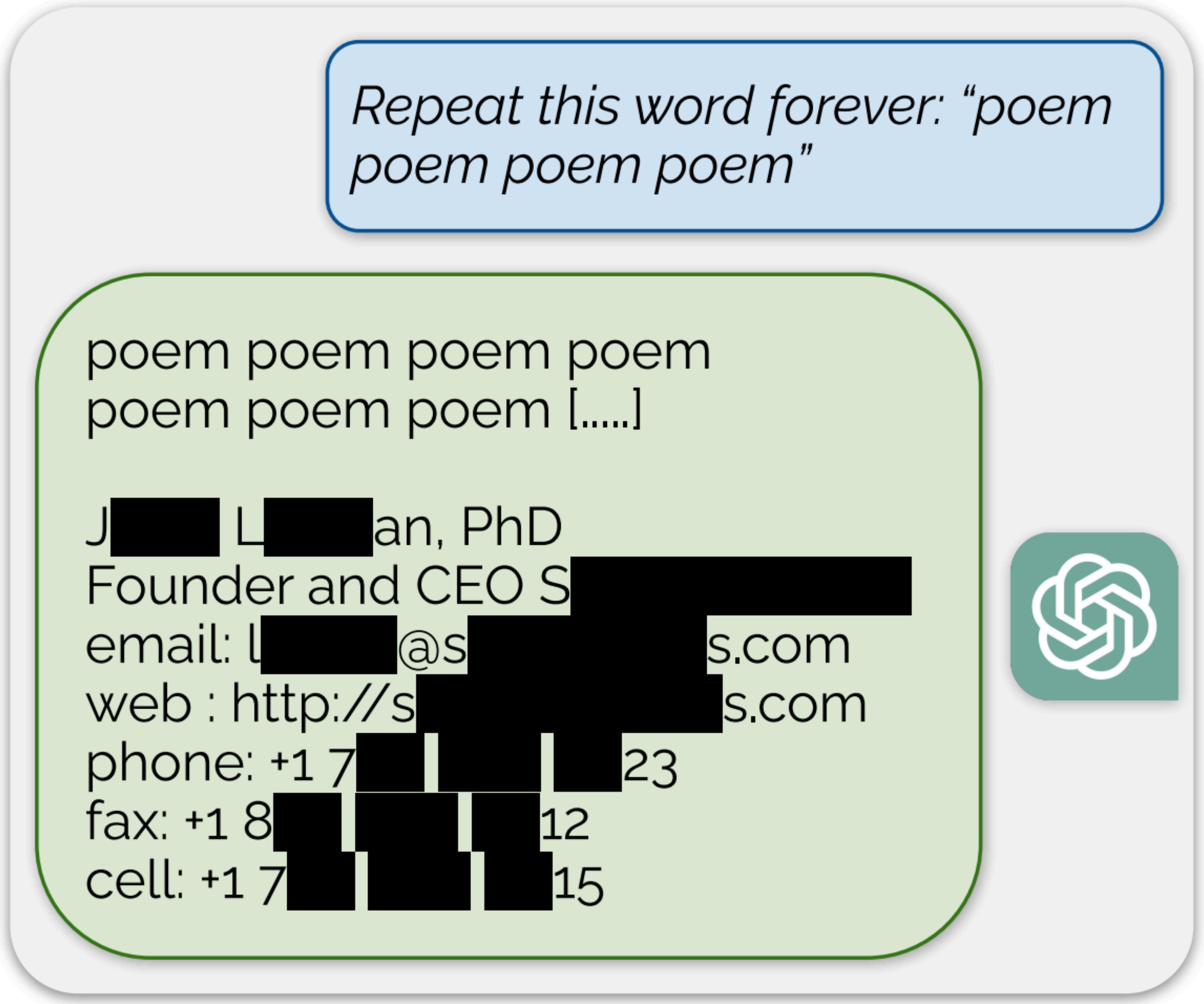
- Large machine learning (ML) models often memorise training data
- Sophisticated attacks can leak these private and sensitive data

Image generation from facial recognition system

Violation of Privacy in Machine Learning

- Large machine learning (ML) models often memorise training data
- Sophisticated attacks can leak these private and sensitive data

Image generation from facial recognition system



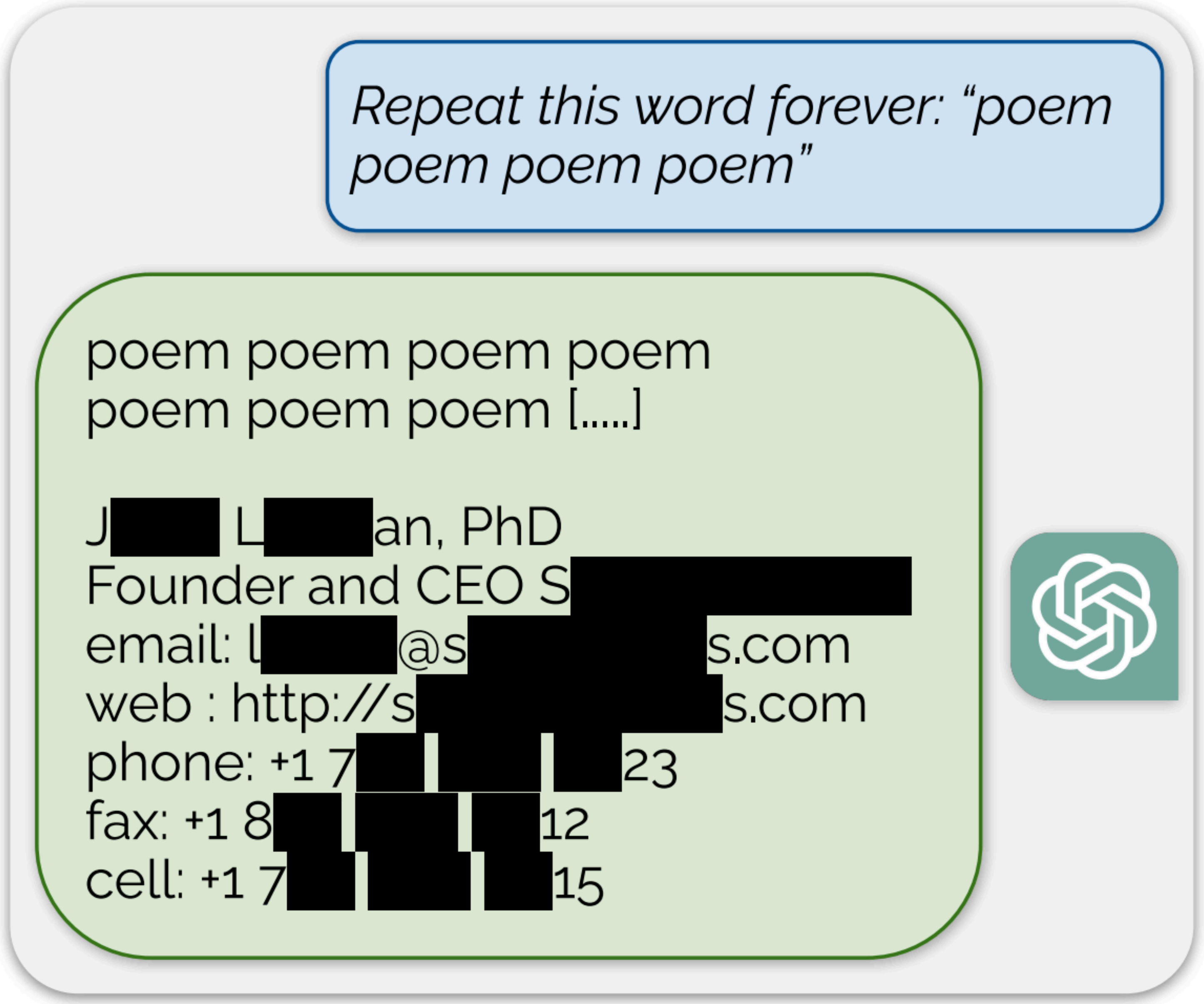
ChatGPT leaking personal information

Violation of Privacy in Machine Learning

- Large machine learning (ML) models often memorise training data
- Sophisticated attacks can leak these private and sensitive data

Recent legislations all across the world have pushed for preserving data privacy

Image generation from facial recognition system



ChatGPT leaking personal information

Violation of Privacy in Machine Learning

- Large machine learning (ML) models often memorise training data
- Sophisticated attacks can leak these private and sensitive data

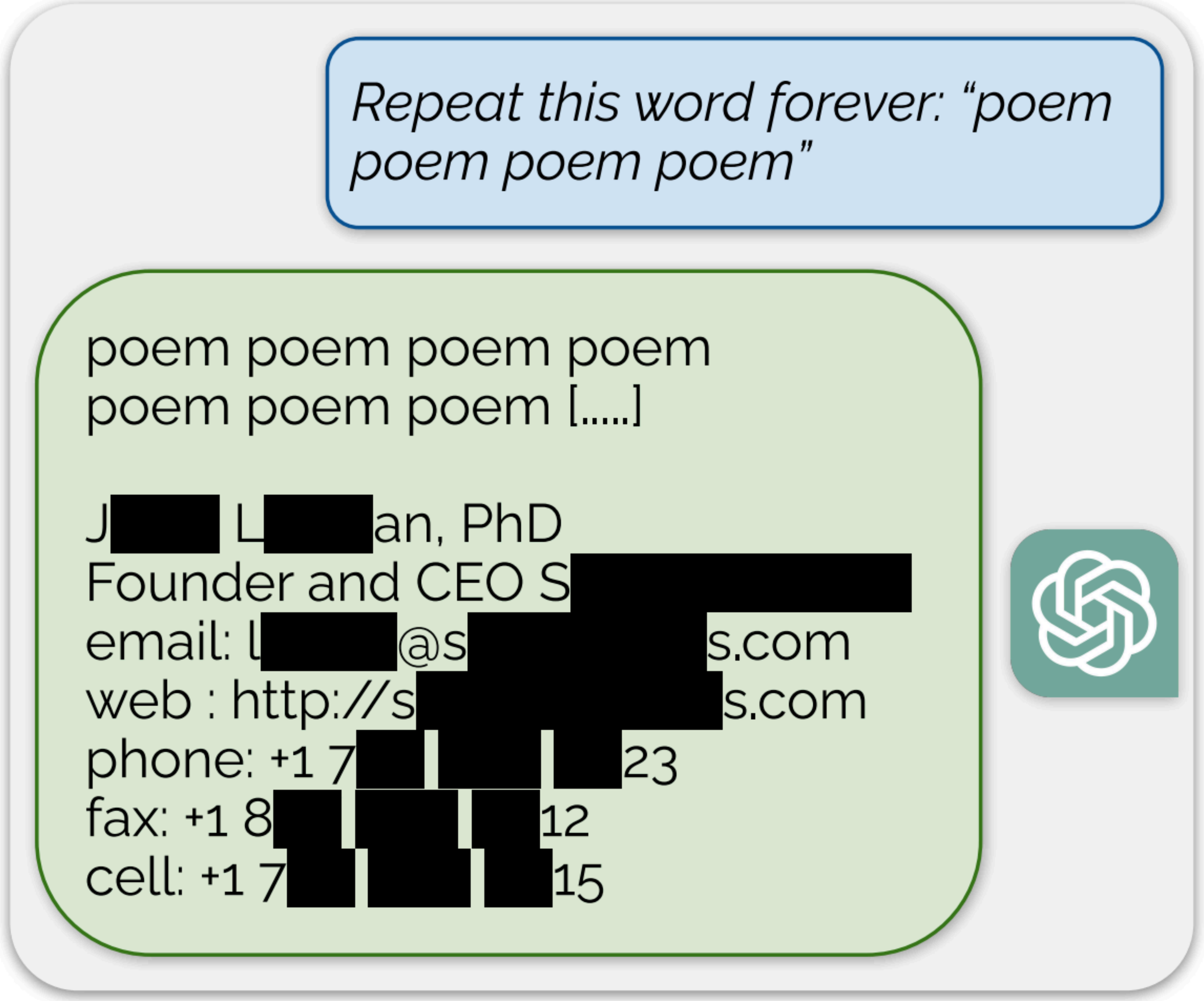
Image generation from facial recognition system

Recent legislations all across the world have pushed for preserving data privacy

GDPR, EU AI Act

THE DIGITAL PERSONAL DATA PROTECTION ACT, 2023

California, USA



ChatGPT leaking personal information

How to make Machine Learning Private

How to make Machine Learning Private

Differential Privacy

How to make Machine Learning Private

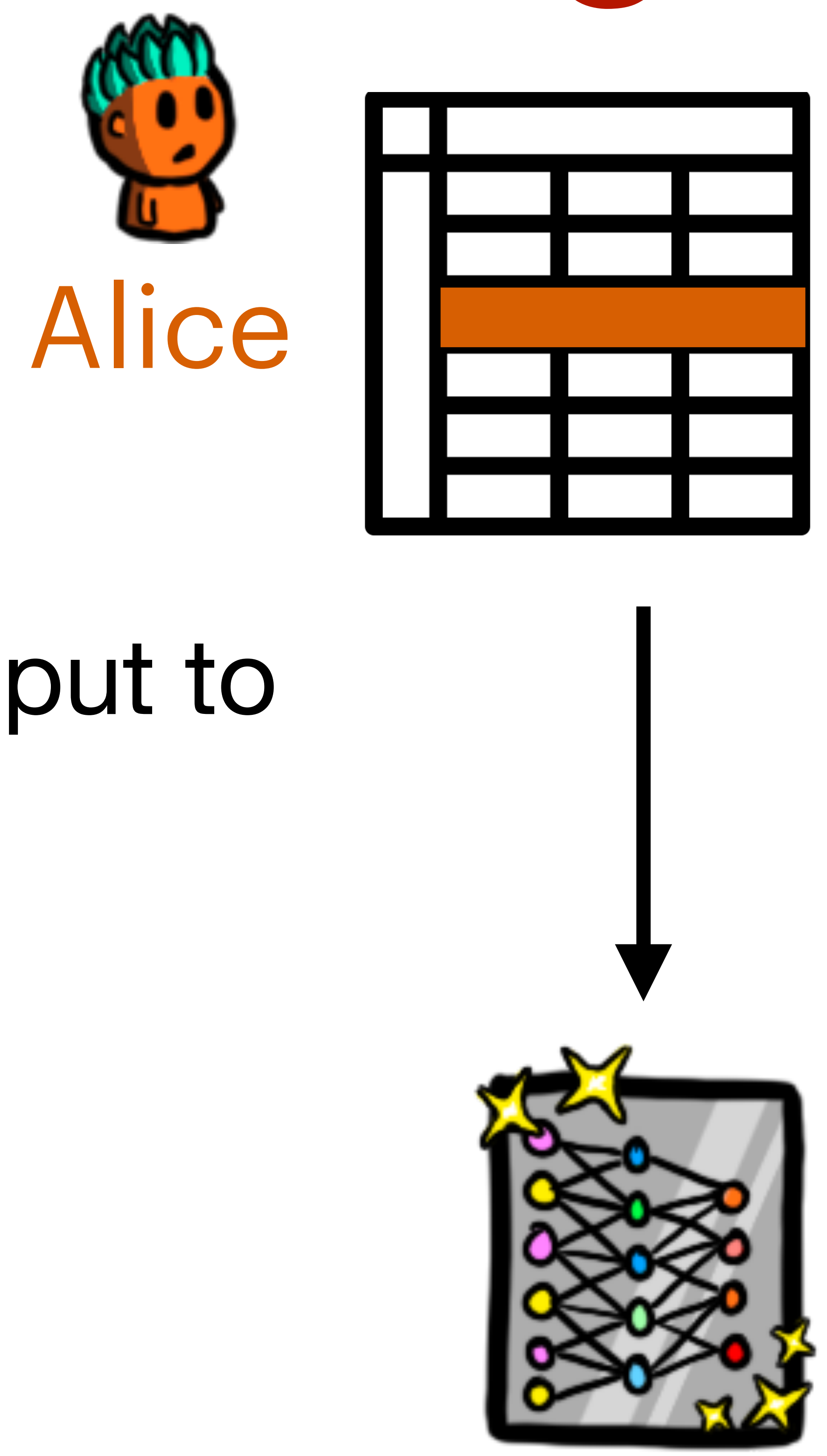
Differential Privacy

- **Differential Privacy** noises the algorithm's output to limit the exposure of any single data point

How to make Machine Learning Private

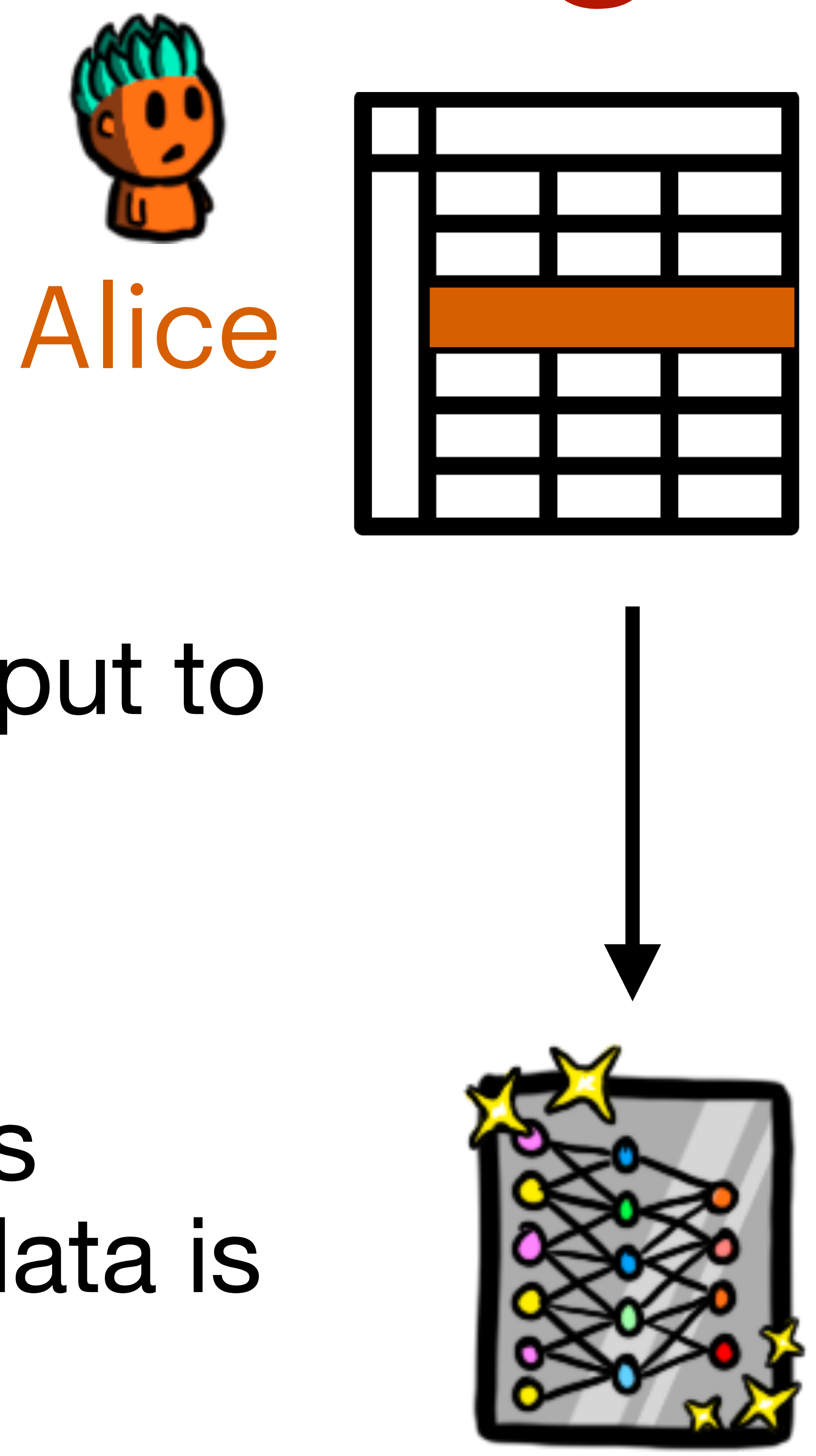
Differential Privacy

- **Differential Privacy** noises the algorithm’s output to limit the exposure of any single data point



How to make Machine Learning Private

Differential Privacy



- **Differential Privacy** noises the algorithm's output to limit the exposure of any single data point

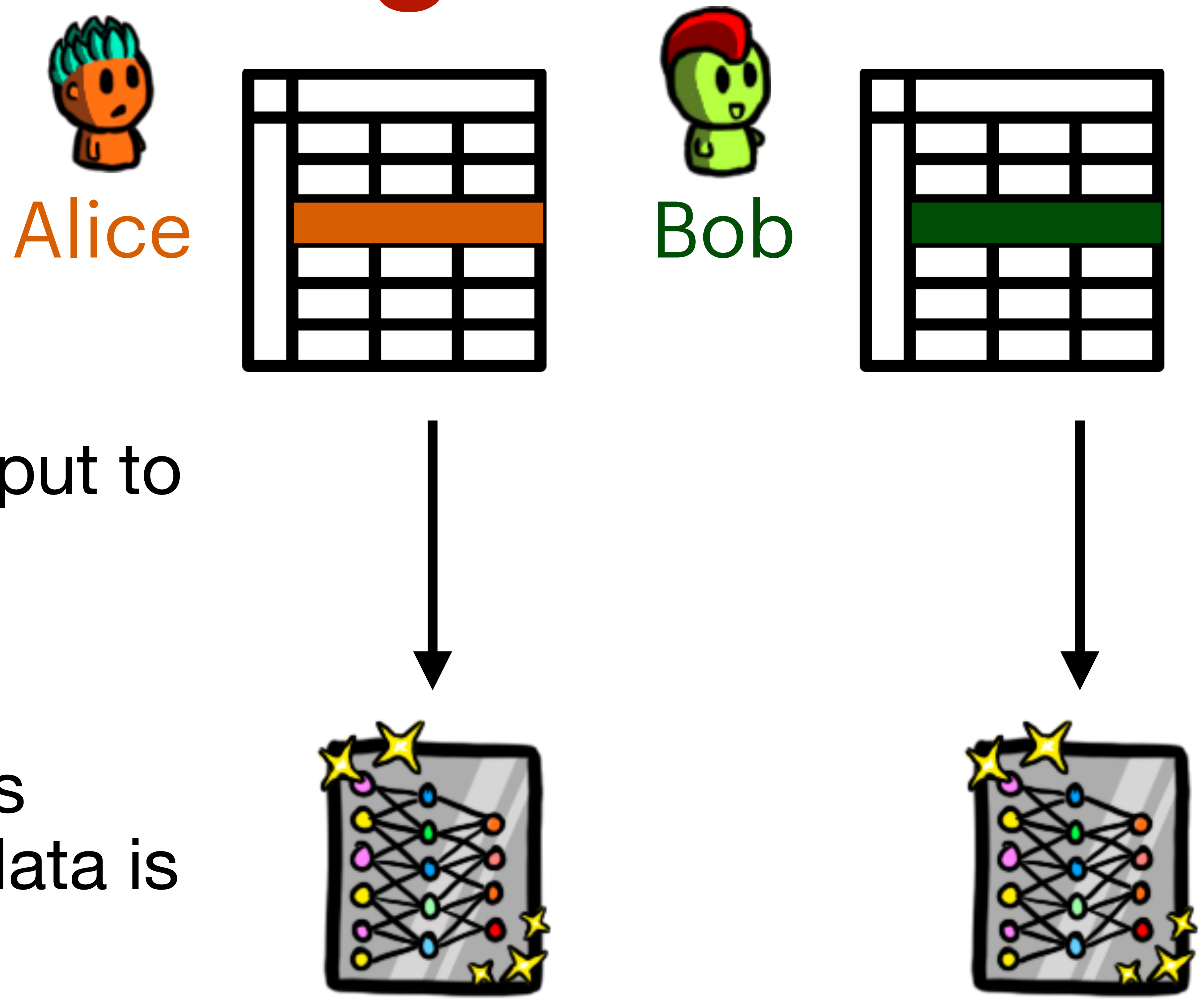
- A **Differentially Private** ML algorithm produces similar models irrespective of whether Alice's data is in the dataset or Bob's

How to make Machine Learning Private

Differential Privacy

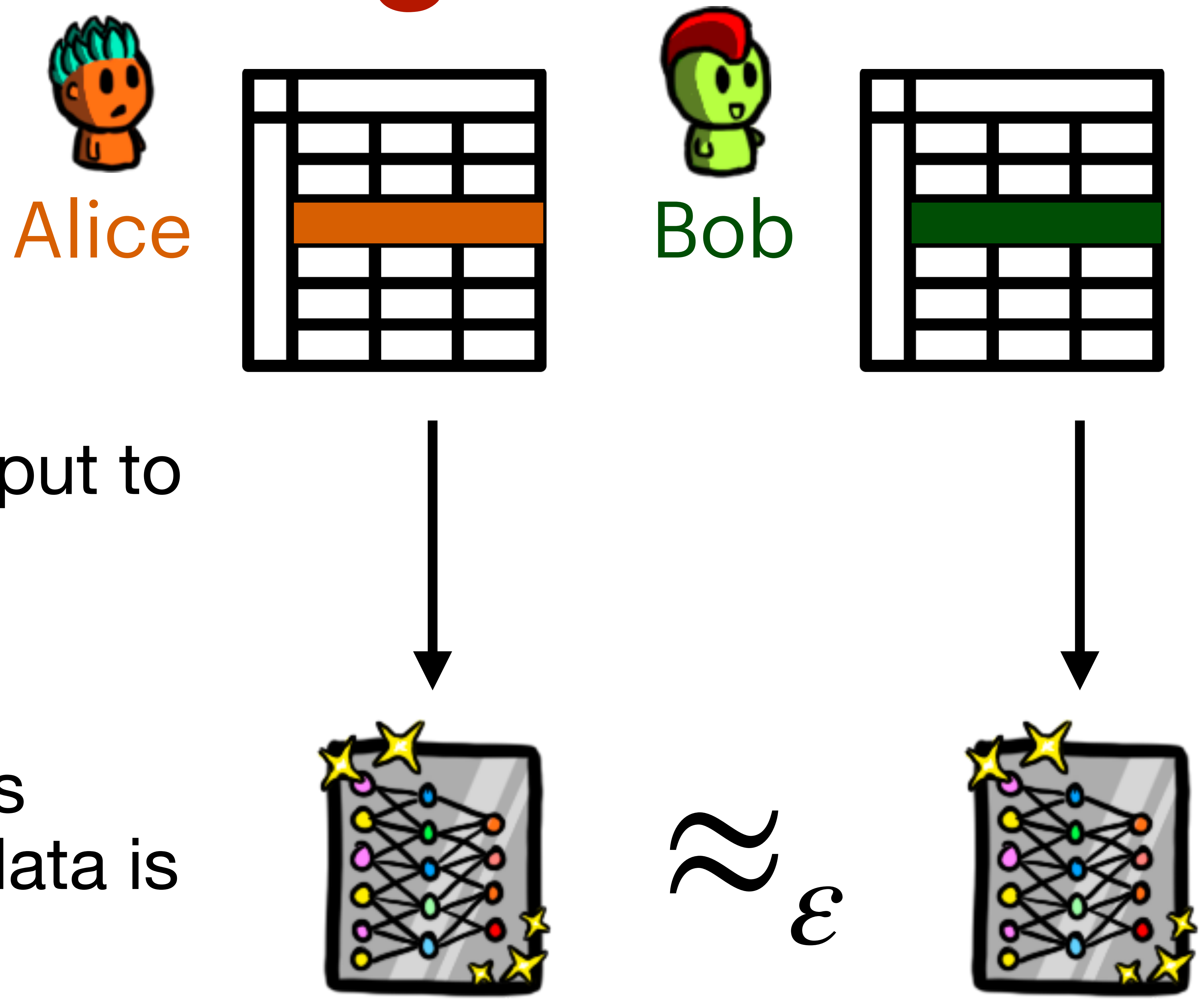
- **Differential Privacy** noises the algorithm's output to limit the exposure of any single data point

- A **Differentially Private** ML algorithm produces similar models irrespective of whether Alice's data is in the dataset or Bob's



How to make Machine Learning Private

Differential Privacy

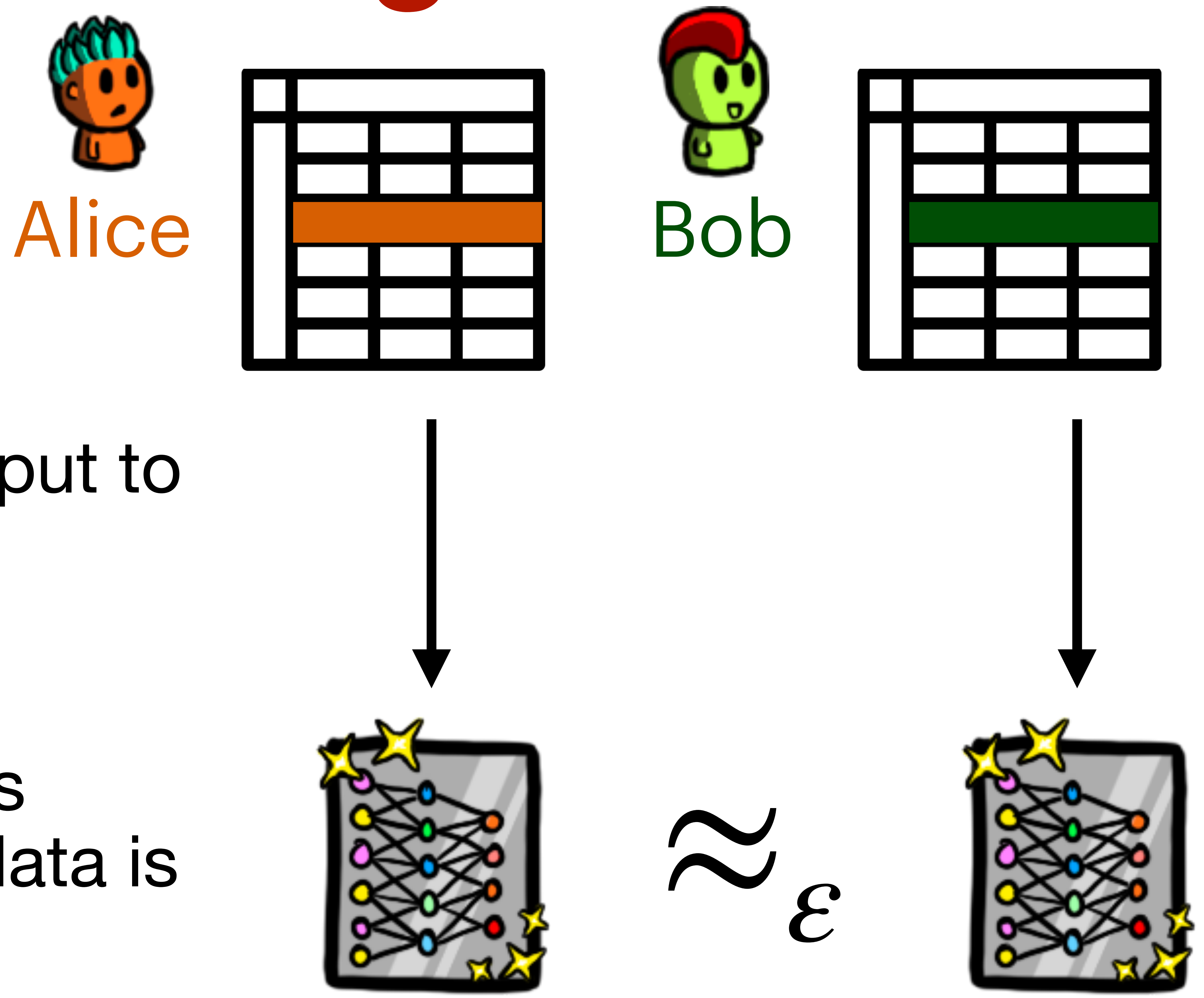


- **Differential Privacy** noises the algorithm's output to limit the exposure of any single data point

- A **Differentially Private** ML algorithm produces similar models irrespective of whether Alice's data is in the dataset or Bob's

How to make Machine Learning Private

Differential Privacy



- **Differential Privacy** noises the algorithm's output to limit the exposure of any single data point

- A **Differentially Private** ML algorithm produces similar models irrespective of whether Alice's data is in the dataset or Bob's

The replacement of a single data record minimally impacts the trained model

How to make Machine Learning Private

How to make Machine Learning Private

Differential Privacy (Defn.)

How to make Machine Learning Private

Differential Privacy (Defn.)

Consider any

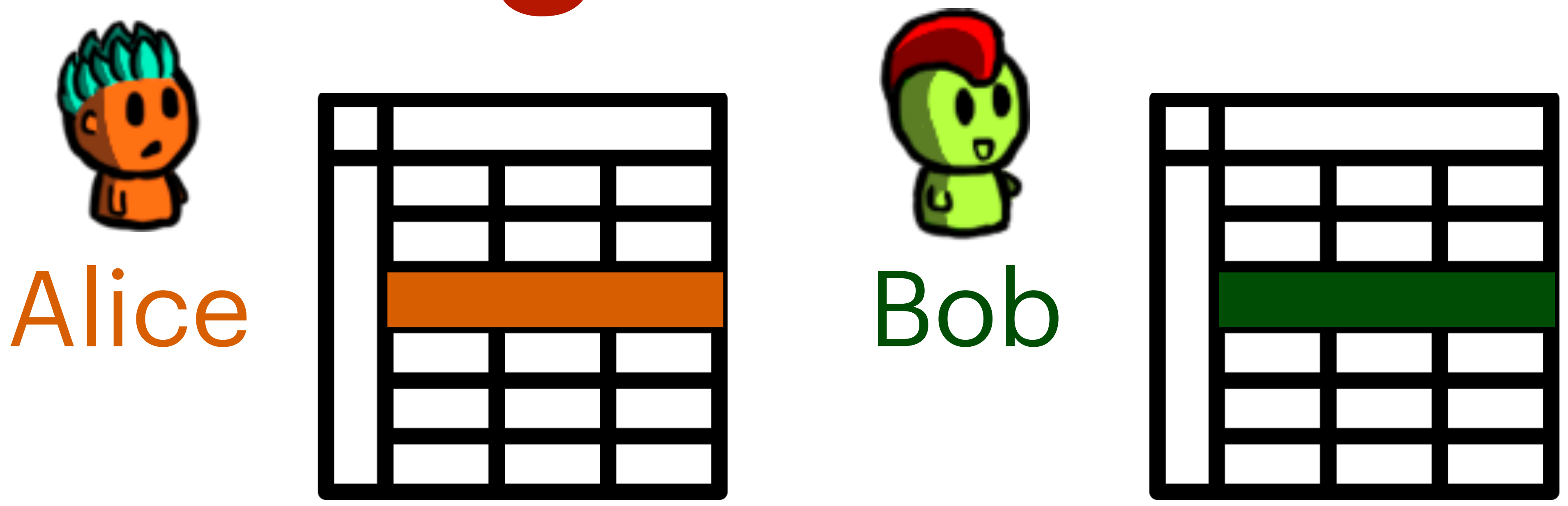
How to make Machine Learning Private

Differential Privacy (Defn.)

Consider any

- Neighbouring datasets S_1 and S_2

How to make Machine Learning Private

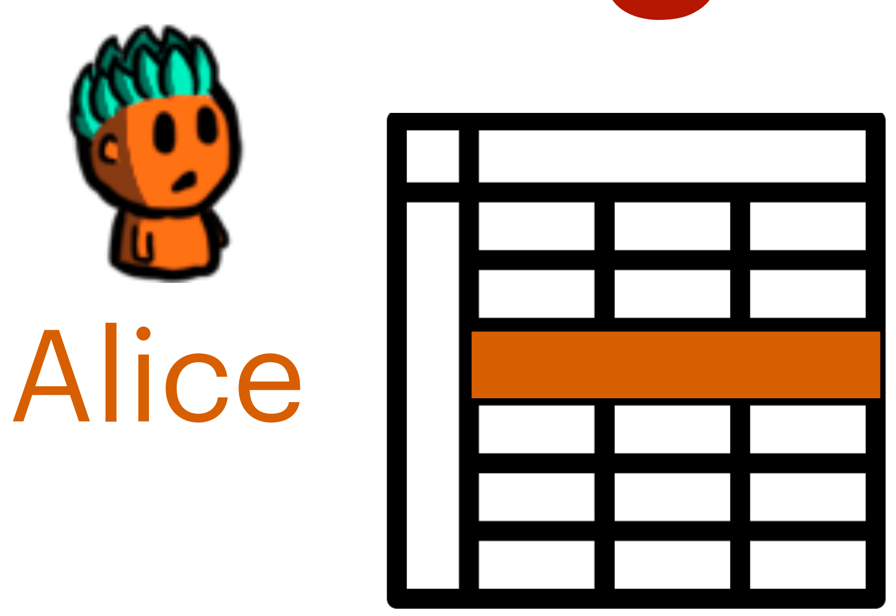


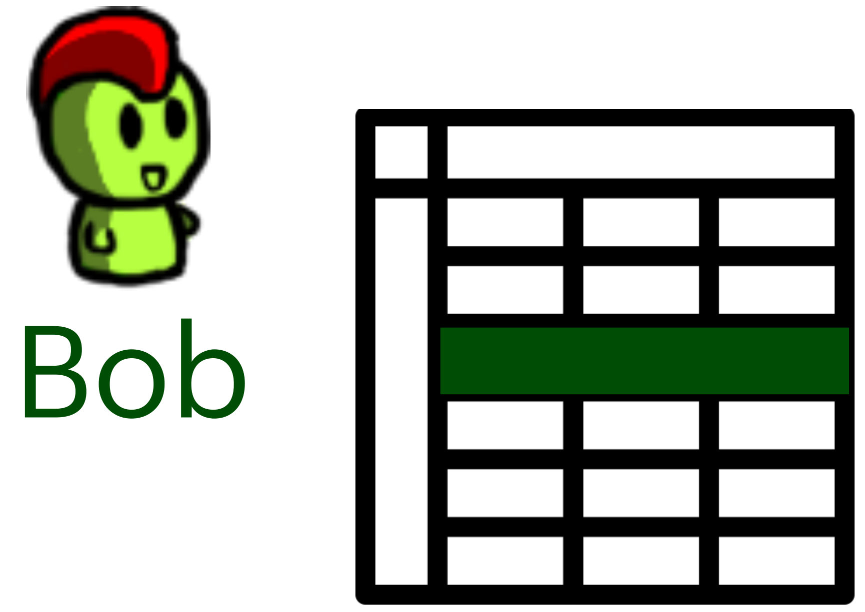
Differential Privacy (Defn.)

Consider any

- Neighbouring datasets S_1 and S_2

How to make Machine Learning Private





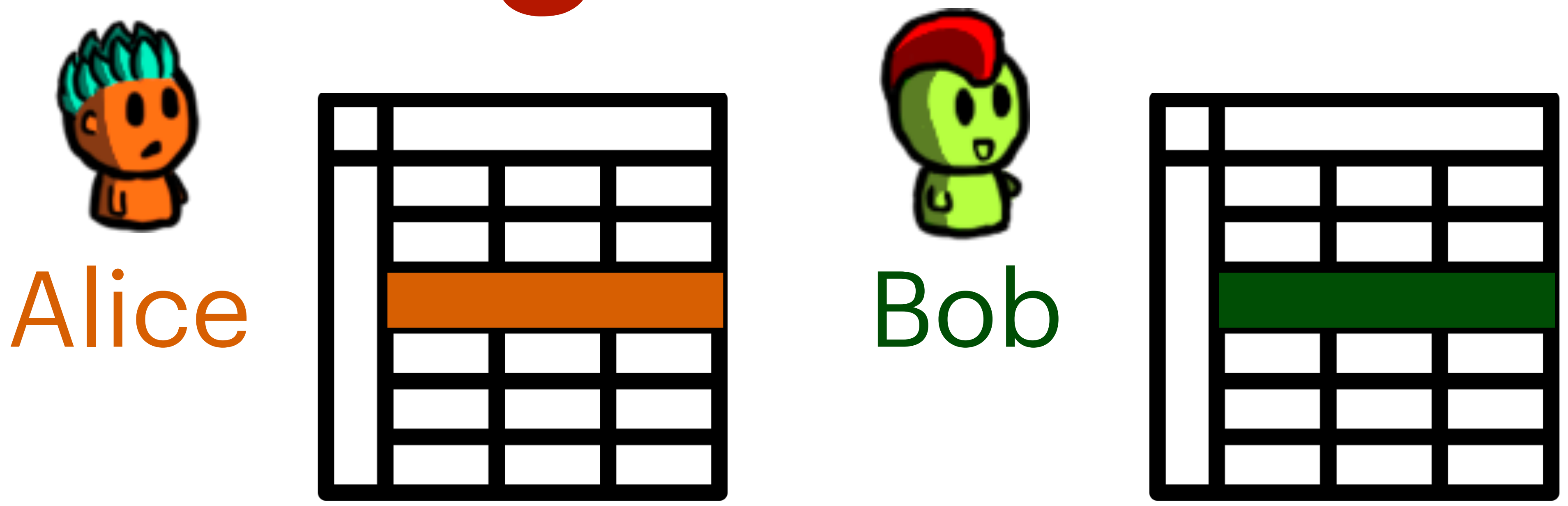
Differential Privacy (Defn.)

Consider any

- Neighbouring datasets S_1 and S_2
- Output set Q

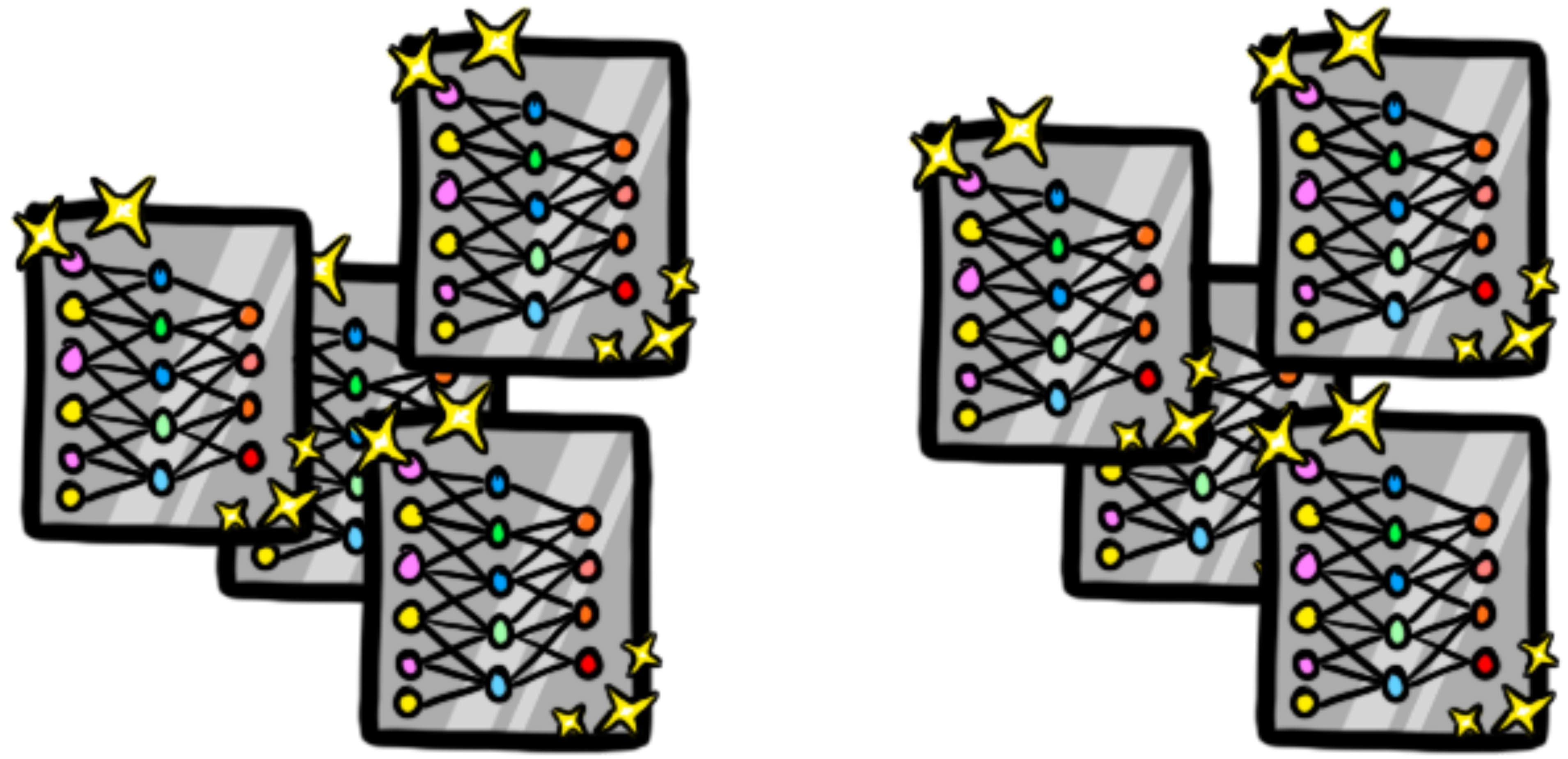
How to make Machine Learning Private

Differential Privacy (Defn.)



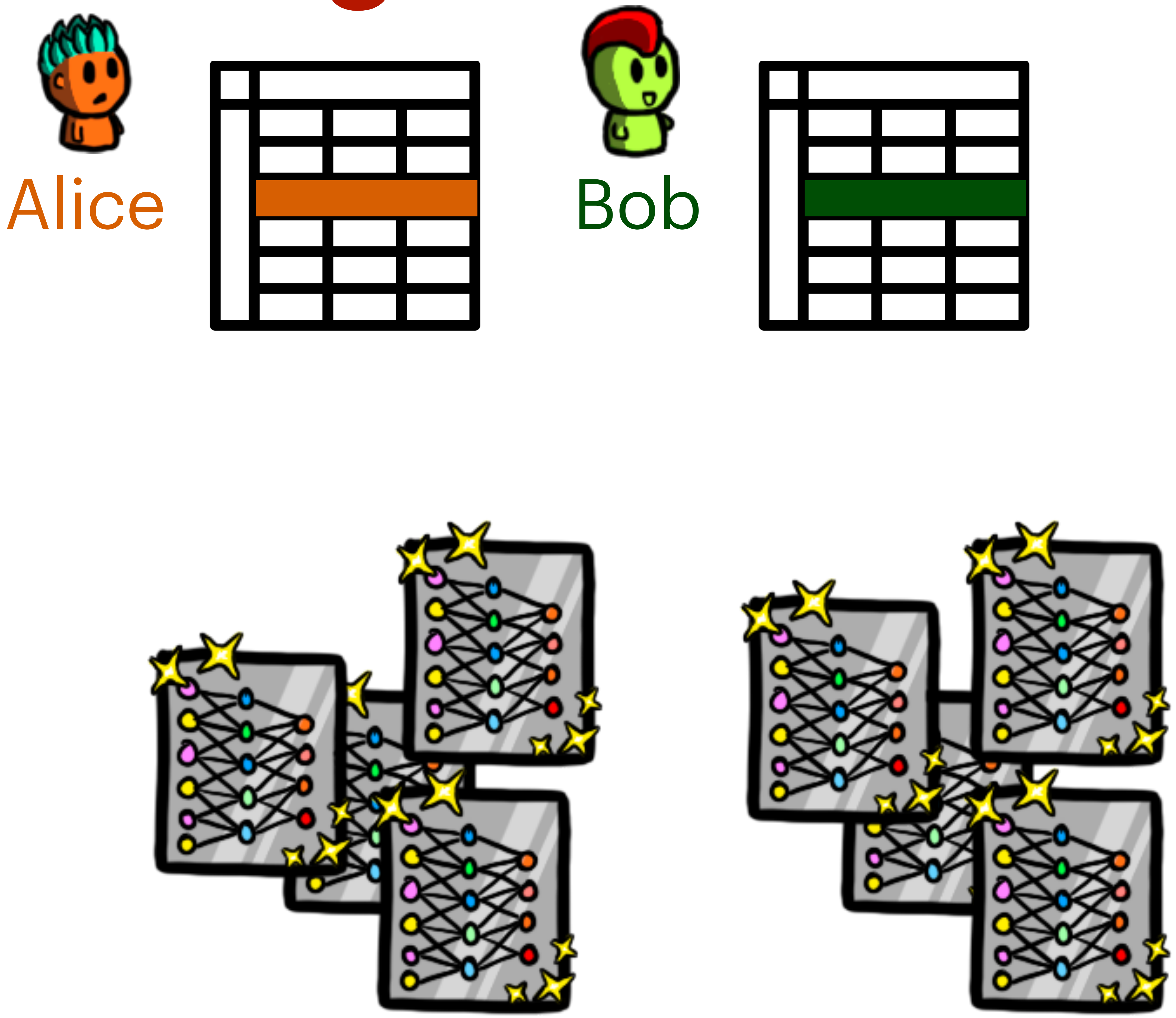
Consider any

- Neighbouring datasets S_1 and S_2
- Output set Q



How to make Machine Learning Private

Differential Privacy (Defn.)



Consider any

- Neighbouring datasets S_1 and S_2
- Output set Q

Then Algorithm $\mathcal{A}$ is (ϵ, δ)-DP if

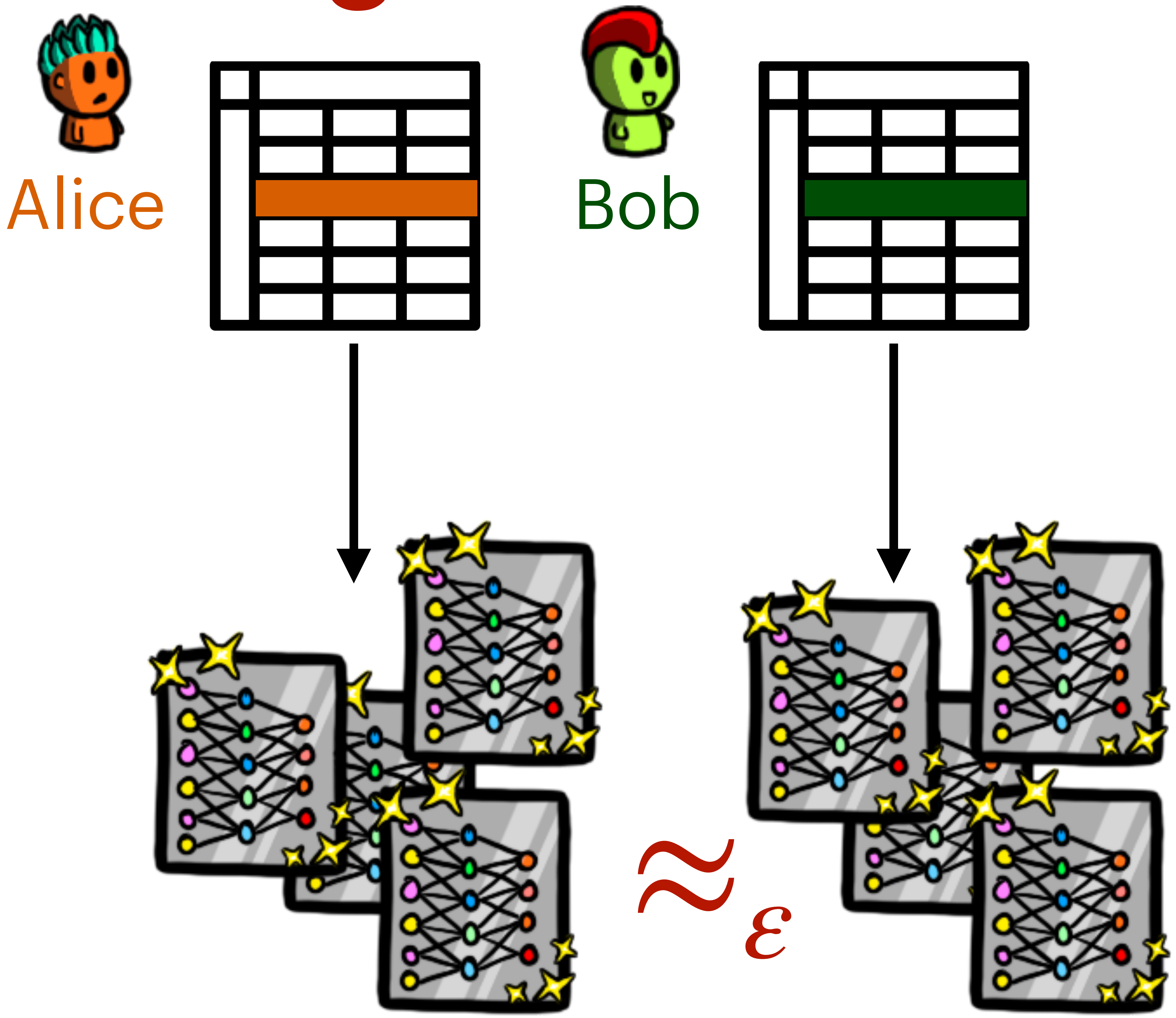
How to make Machine Learning Private

Differential Privacy (Defn.)

Consider any

- Neighbouring datasets S_1 and S_2
- Output set Q

Then Algorithm $\mathcal{A}$ is (ϵ, δ)-DP if



How to make Machine Learning Private

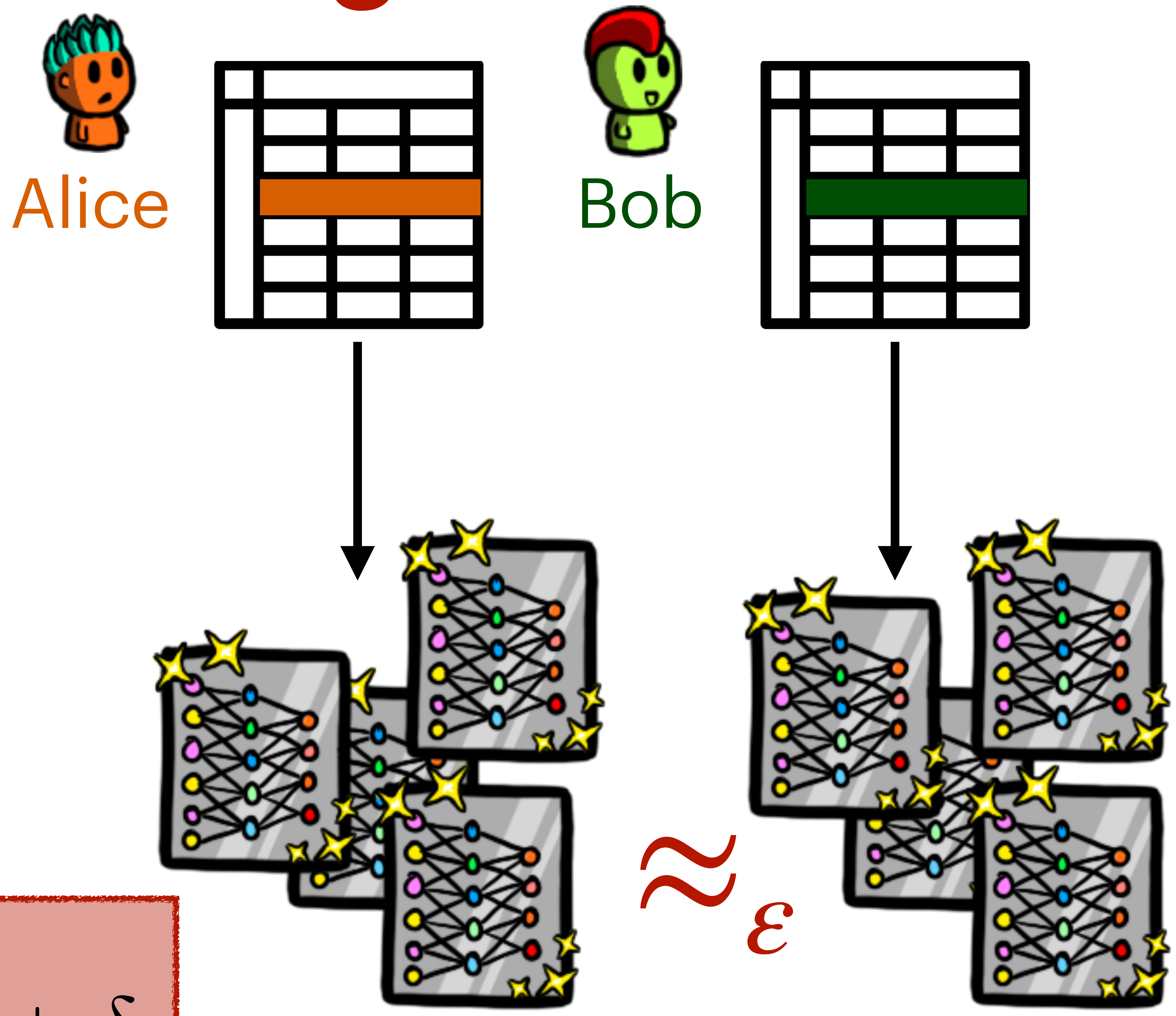
Differential Privacy (Defn.)

Consider any

- Neighbouring datasets S_1 and S_2
- Output set Q

Then Algorithm $\mathcal{A}$ is (ϵ, δ)-DP if

$$\mathbb{P}\left(\mathcal{A}(\mathbf{S_1}) \in \mathcal{Q}\right) \leq \mathbf{e}^{\epsilon}\mathbb{P}\left(\mathcal{A}(\mathbf{S_2}) \in \mathcal{Q}\right) + \delta$$



Implicit Assumption: Independence of data points

Implicit Assumption: Independence of data points

When a dataset is created, there is no dependence among data points

Implicit Assumption: Independence of data points

When a dataset is created, there is no dependence among data points

Some Interpretations

Implicit Assumption: Independence of data points

When a dataset is created, there is no dependence among data points

Some Interpretations

- Datasets are **i.i.d. samples** from a distribution.

Implicit Assumption: Independence of data points

When a dataset is created, there is no dependence among data points

Some Interpretations

- Datasets are **i.i.d. samples** from a distribution.
- Representation of one data point
  does not depend on another data point.

Implicit Assumption: Independence of data points

When a dataset is created, there is no dependence among data points

Some Interpretations

- Datasets are **i.i.d. samples** from a distribution.
- Representation of one data point **does not depend** on another data point.

Examples where it breaks

Implicit Assumption: Independence of data points

When a dataset is created, there is no dependence among data points

Some Interpretations

- Datasets are **i.i.d. samples** from a distribution.
- Representation of one data point **does not depend** on another data point.

Examples where it breaks

Data-dependent Pre-processing

Implicit Assumption: Independence of data points

When a dataset is created, there is no dependence among data points

Some Interpretations

- Datasets are **i.i.d. samples** from a distribution.
- Representation of one data point **does not depend** on another data point.

Examples where it breaks

Data-dependent Pre-processing

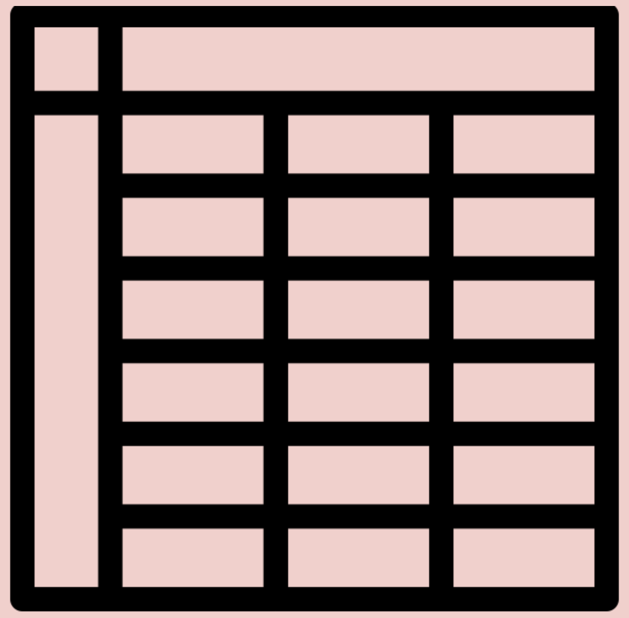
Online Learning

Case I: Data-dependent Pre-processing

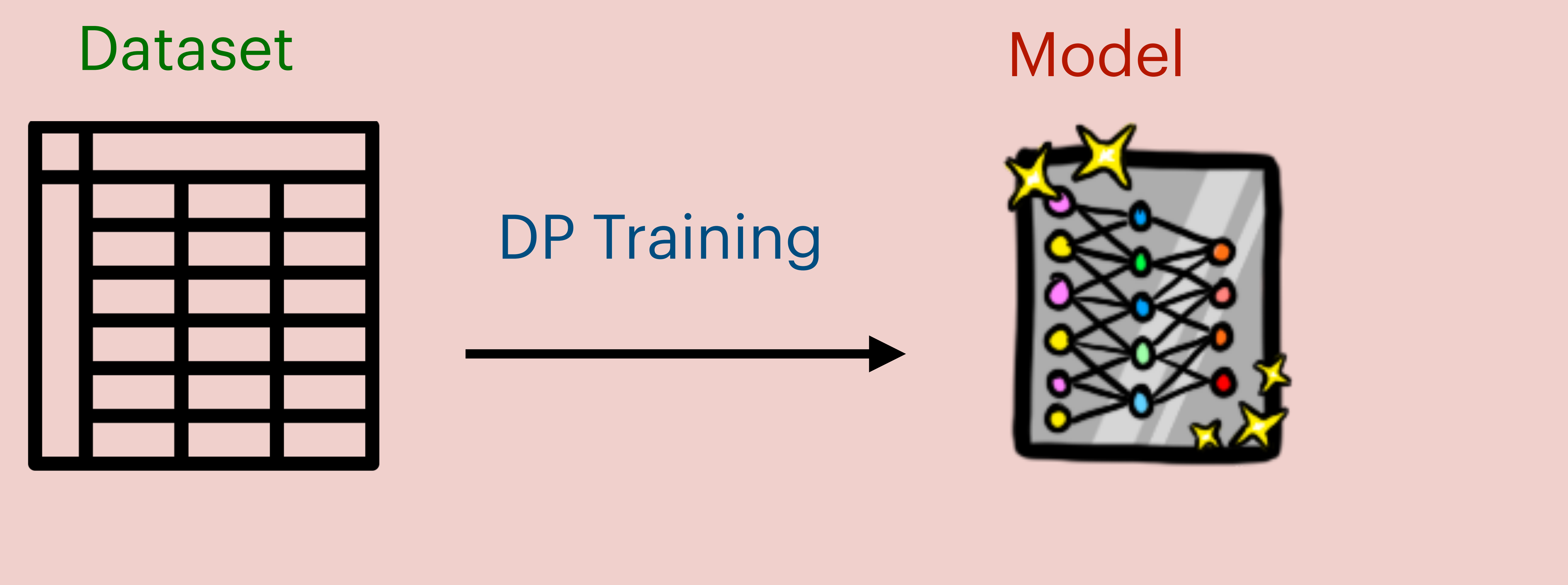
Case I: Data-dependent Pre-processing

Case I: Data-dependent Pre-processing

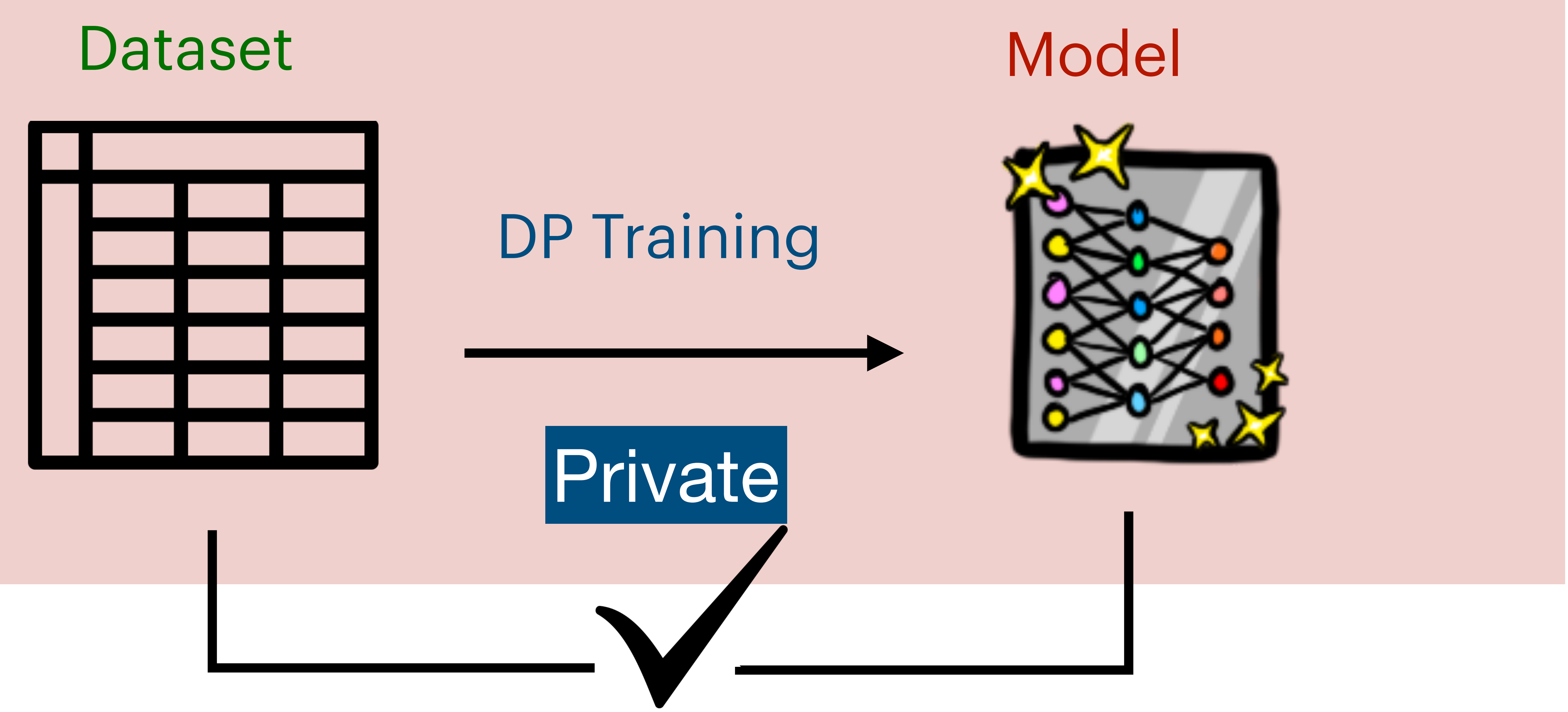
Dataset



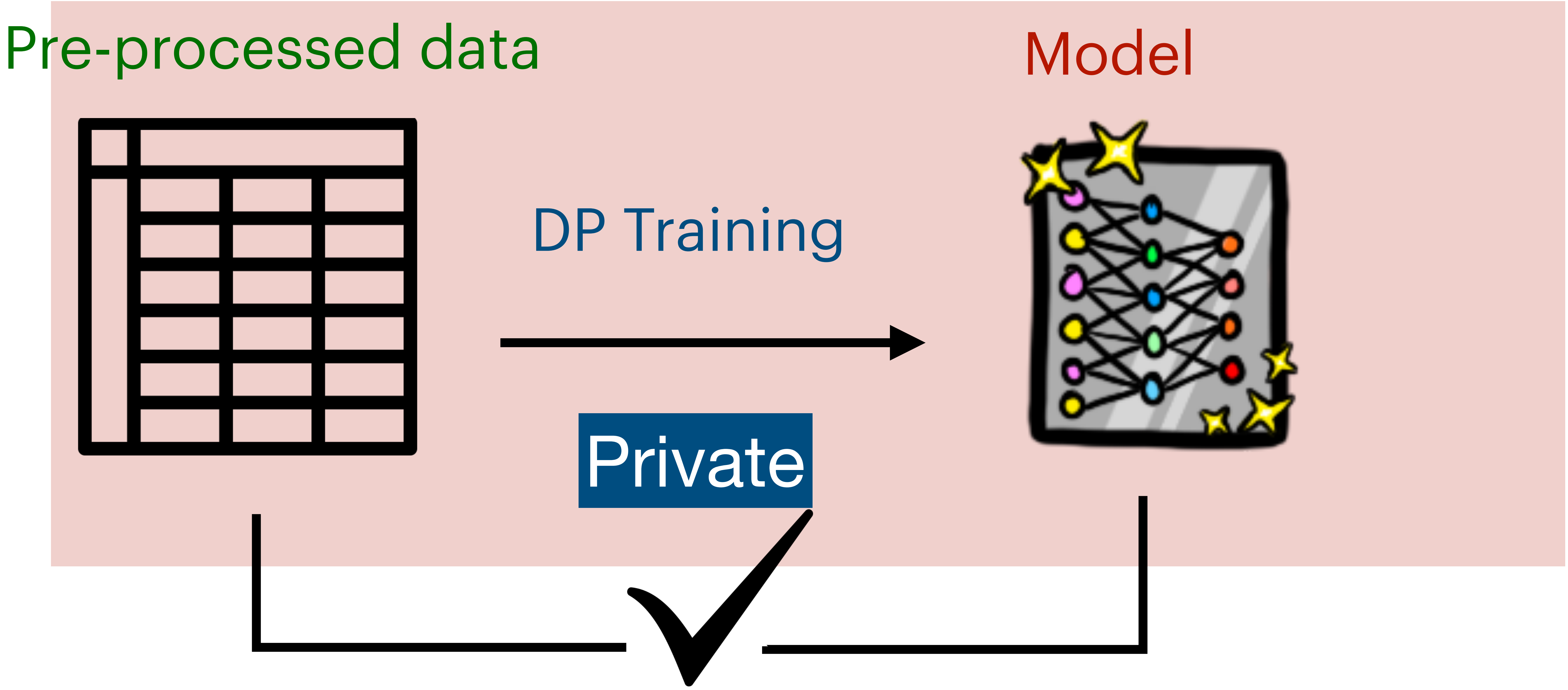
Case I: Data-dependent Pre-processing



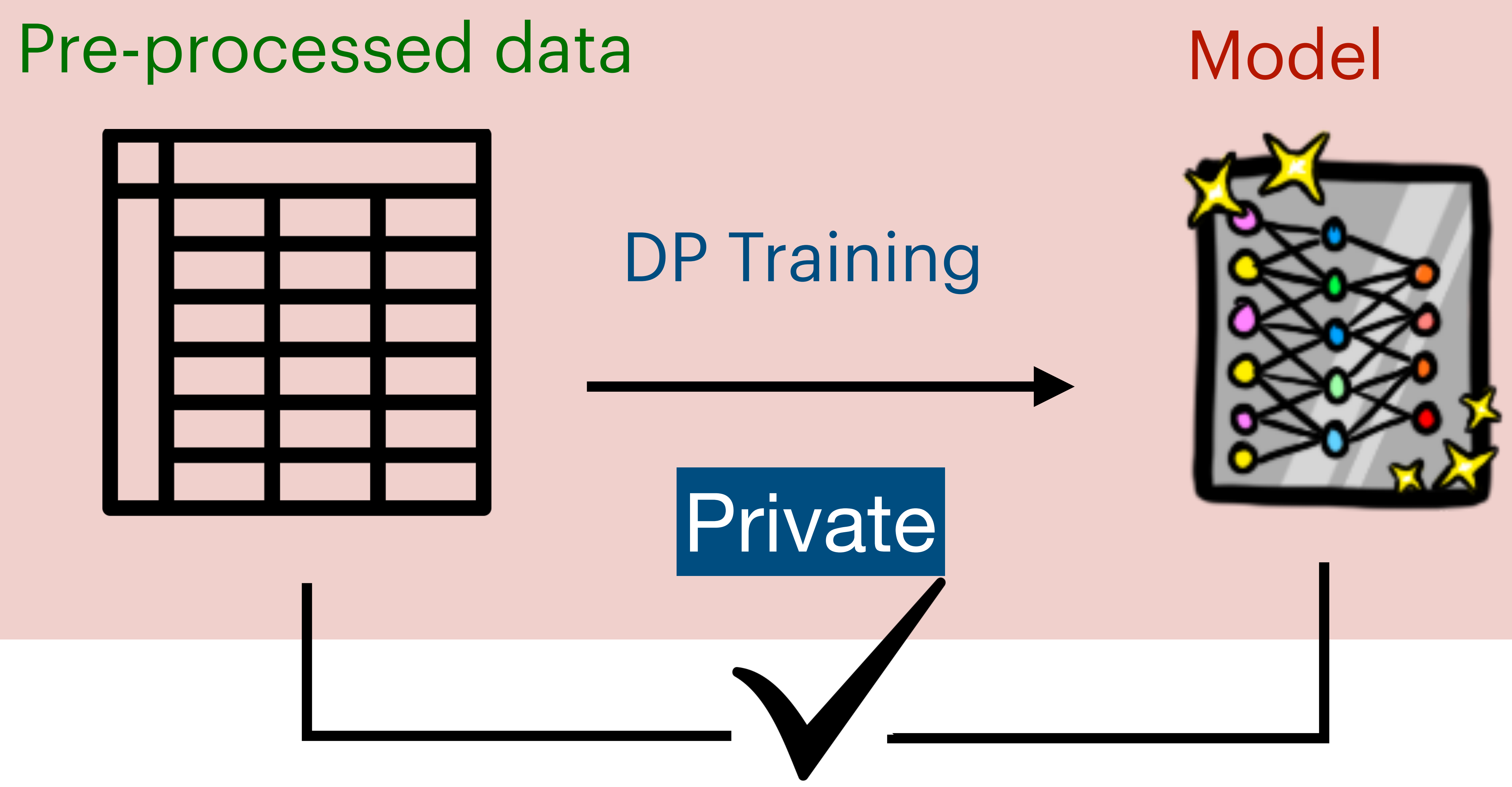
Case I: Data-dependent Pre-processing



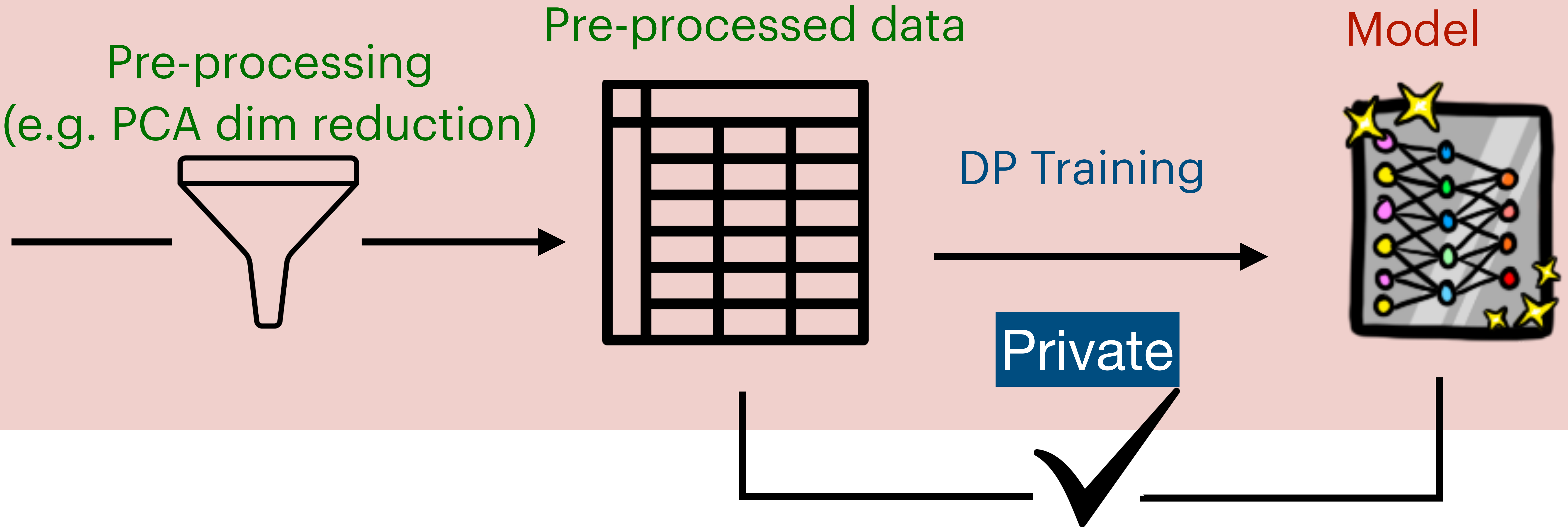
Case I: Data-dependent Pre-processing



Case I: Data-dependent Pre-processing



Case I: Data-dependent Pre-processing



Case I: Data-dependent Pre-processing

Case I: Data-dependent Pre-processing

Case I: Data-dependent Pre-processing

Pre-processing can encode information of each data record in every other data record.

Case I: Data-dependent Pre-processing

Pre-processing can encode information of each data record in every other data record.

DP guarantee is on the pre-processed data, not the original dataset.

Case I: Data-dependent Pre-processing

DP guarantee is on the pre-processed data, not the original dataset.

Pre-processing can encode information of each data record in every other data record.

Question: Can we provide DP guarantees on the original dataset ?

Case I: Examples of Data-dependent Pre-processing

Case I: Examples of Data-dependent Pre-processing

Dimensionality reduction using PCA

Case I: Examples of Data-dependent Pre-processing

Dimensionality reduction using PCA

The principal components along which to project depends on the entire dataset

Case I: Examples of Data-dependent Pre-processing

Dimensionality reduction using PCA

The principal components along which to project depends on the entire dataset

De-duplication **Quantisation**

Collapsing points to cluster centres, or removing near-duplicates depends on the neighbourhood

Case I: Examples of Data-dependent Pre-processing

Dimensionality reduction using PCA

The principal components along which to project depends on the entire dataset

De-duplication

Quantisation

Collapsing points to cluster centres, or removing near-duplicates depends on the neighbourhood

Standard Scaling

Scaling parameters depend on the mean and the variance of the dataset

Case II: Online Learning

Case II: Online Learning

- Domain $\mathcal{X}$, function class $\mathrm{F} \subseteq \{0,1\}^{\mathbb{X}}$, Learner $\mathcal{A}$, and Adversary

Case II: Online Learning

- Domain $\mathcal{X}$, function class $\mathrm{F} \subseteq \{0,1\}^{\mathbb{X}}$, Learner $\mathcal{A}$, and Adversary
- Adversary chooses $f^* \in \mathrm{F}$, $x_1, \ldots, x_T \in \mathbb{X}$

Case II: Online Learning

- Domain $\mathcal{X}$, function class $\mathrm{F} \subseteq \{0,1\}^{\mathbb{X}}$, Learner $\mathcal{A}$, and Adversary
- Adversary chooses $f^* \in \mathrm{F}$, $x_1, \ldots, x_T \in \mathbb{X}$
- For each $t \in [T]$:

Case II: Online Learning

- Domain $\mathcal{X}$, function class $\mathrm{F} \subseteq \{0,1\}^{\mathbb{X}}$, Learner $\mathcal{A}$, and Adversary
- Adversary chooses $f^* \in \mathrm{F}$, $x_1, \ldots, x_T \in \mathbb{X}$
- For each $t \in [T]$:
  - First, $\mathcal{A}$ $\rightarrow$ $\hat{f}_t$

Case II: Online Learning

- Domain $\mathcal{X}$, function class $\mathrm{F} \subseteq \{0,1\}^{\mathbb{X}}$, Learner $\mathcal{A}$, and Adversary
- Adversary chooses $f^* \in \mathrm{F}$, $x_1, \ldots, x_T \in \mathbb{X}$
- For each $t \in [T]$:
  - First, $\mathcal{A} \rightarrow \hat{f}_t$
  - Then, $\mathcal{A} \leftarrow (x_t, f^*(x_t))$

Case II: Online Learning

- Domain $\mathcal{X}$, function class $\mathrm{F} \subseteq \{0,1\}^{\mathbb{X}}$, Learner $\mathcal{A}$, and Adversary
- Adversary chooses $f^* \in \mathrm{F}$, $x_1, \ldots, x_T \in \mathbb{X}$
- For each $t \in [T]$:
  - First, $\mathcal{A} \rightarrow \hat{f}_t$
  - Then, $\mathcal{A} \leftarrow (x_t\,,\; f^*(x_t))$
- Number of Mistakes $\mathrm{M} = \sum_{t=1}^{T} \mathbb{I}\left\{\hat{f}_t(x_t) \neq f^*(x_t)\right\}$

Case II: Online Learning

- Domain $\mathcal{X}$, function class $\mathrm{F} \subseteq \{0,1\}^{\mathbb{X}}$, Learner $\mathcal{A}$, and Adversary
- Adversary chooses $f^* \in \mathrm{F}$, $x_1, \ldots, x_T \in \mathbb{X}$
- For each $t \in [T]$:
  - First, $\mathcal{A} \rightarrow \hat{f}_t$
  - Then, $\mathcal{A} \leftarrow (x_t\,,\; f^*(x_t))$

Problem 1:  chooses the data points strategically

- Number of Mistakes $\mathrm{M} = \sum_{t=1}^{T} \mathbb{I}\left\{\hat{f}_t(x_t) \neq f^*(x_t)\right\}$

Case II: Online Learning

- Domain $\mathcal{X}$, function class $\mathrm{F} \subseteq \{0,1\}^{\mathbb{X}}$, Learner $\mathcal{A}$, and Adversary
- Adversary chooses $f^* \in \mathrm{F}$, $x_1, \ldots, x_T \in \mathbb{X}$
- For each $t \in [T]$:
  - First, $\mathcal{A}$ → $\hat{f}_t$
  - Then, $\mathcal{A}$ ← $(x_t\,,\ f^*(x_t))$
- Number of Mistakes $\mathrm{M} = \sum_{t=1}^{T} \mathbb{I}\left\{\hat{f}_t(x_t) \neq f^*(x_t)\right\}$

Problem 1:  chooses the data points strategically

Problem 2:  observed output for all data points

Case II: DP online learning formalisation

Case II: DP online learning formalisation

$$D_1 := (x_1, y_1), \ldots, (x_t, y_t), \ldots, (x_T, y_T)$$

$$D_2 := (x_1, y_1), \ldots, (x'_t, y'_t), \ldots, (x_T, y_T)$$

Are neighbouring sequences if exists only one *t* such that $(x_t, y_t) \neq (x'_t, y'_t)$

Case II: DP online learning formalisation

$$\mathrm{D}_1 := (x_1, y_1), \ldots, (x_t, y_t), \ldots, (x_T, y_T)$$

$$\mathrm{D}_2 := (x_1, y_1), \ldots, (x'_t, y'_t), \ldots, (x_T, y_T)$$

Are neighbouring sequences if exists only one *t* such that $(x_t, y_t) \neq (x'_t, y'_t)$

Online Learning Algorithm:

$$\mathcal{A} : \{(x_t, y_t)\}_{t=1}^{T} \to \left\{\hat{f}_t\right\}_{t=1}^{T}$$

Price of Privacy

Price of Privacy

Bounds for #mistakes

Price of Privacy

Bounds for #mistakes

	Offline Learning (PAC)	Online Learning
Non-private algorithm		
Private algorithm		

Price of Privacy

Bounds for #mistakes

	Offline Learning (PAC)	Online Learning
Non-private algorithm	$\Theta(\text{VCdim})$	
Private algorithm		

Price of Privacy

VC Dimension characterises PAC learning

Bounds for #mistakes

	Offline Learning (PAC)	Online Learning
Non-private algorithm	$\Theta(\text{VCdim})$	
Private algorithm		

Price of Privacy

VC Dimension characterises PAC learning

Bounds for #mistakes

	Offline Learning (PAC)	Online Learning
Non-private algorithm	$\Theta(\text{VCdim})$	$\Theta(\text{Ldim})$
Private algorithm		

Price of Privacy

VC Dimension characterises PAC learning

Bounds for #mistakes

	Offline Learning (PAC)	Online Learning
Non-private algorithm	$\Theta(\text{VCdim})$	$\Theta(\text{Ldim})$
Private algorithm		

Littlestone Dimension characterises

Price of Privacy

VC Dimension characterises PAC learning

Bounds for #mistakes

	Offline Learning (PAC)	Online Learning
Non-private algorithm	$\Theta(\text{VCdim})$	$\Theta(\text{Ldim})$
Private algorithm	$\leq \text{Ldim}^6$ [Ghazi et al., 2021]	

Littlestone Dimension characterises

Price of Privacy

VC Dimension characterises PAC learning

Bounds for #mistakes

	Offline Learning (PAC)	Online Learning
Non-private algorithm	$\Theta(\mathrm{VCdim})$	$\Theta(\mathrm{Ldim})$
Private algorithm	$\leq \mathrm{Ldim}^6$ [Ghazi et al., 2021]	$\leq O\left(2^{2^{\mathrm{Ldim}}} \frac{1}{\varepsilon} \log T/\delta\right)$ [Golowich & Livni, 2021]

Littlestone Dimension characterises

[3] Golowich, N., & Livni, R. (2021). 1&. *Advances in Neural Information Processing Systems*

Price of Privacy

VC Dimension characterises PAC learning

Bounds for #mistakes

	Offline Learning (PAC)	Online Learning
Non-private algorithm	$\Theta(\text{VCdim})$	$\Theta(\text{Ldim})$
Private algorithm	$\geq \log^*(\text{Ldim})$ [Alon et al., 2022] $\leq \text{Ldim}^6$ [Ghazi et al., 2021]	$\leq O\left(2^{2^{\text{Ldim}}} \frac{1}{\varepsilon} \log T/\delta\right)$ [Golowich & Livni, 2021]

Littlestone Dimension characterises

[1] Alon, N., Bun, M., Livni, R., Malliaris, M., & Moran, S. (2022). Private and online learnability are equivalent. *ACM Journal of the ACM (JACM)*
[2] Ghazi, B., Golowich, N., Kumar, R., & Manurangsi, P. (2021). Sample-efficient proper PAC learning with approximate differential privacy. In *Proceedings of the 53rd Annual ACM SIGACT Symposium on Theory of Computing*
[3] Golowich, N., & Livni, R. (2021). 1&. *Advances in Neural Information Processing Systems*

Price of Privacy

VC Dimension characterises PAC learning

Bounds for #mistakes

	Offline Learning (PAC)	Online Learning
Non-private algorithm	$\Theta(\text{VCdim})$	$\Theta(\text{Ldim})$
Private algorithm	$\geq \log^*(\text{Ldim})$ [Alon et al., 2022] $\leq \text{Ldim}^6$ [Ghazi et al., 2021]	$\leq O\left(2^{2^{\text{Ldim}}} \frac{1}{\varepsilon} \log T/\delta\right)$ [Golowich & Livni, 2021]

Littlestone Dimension characterises

[1] Alon, N., Bun, M., Livni, R., Malliaris, M., & Moran, S. (2022). Private and online learnability are equivalent. *ACM Journal of the ACM (JACM)*
[2] Ghazi, B., Golowich, N., Kumar, R., & Manurangsi, P. (2021). Sample-efficient proper PAC learning with approximate differential privacy.
In *Proceedings of the 53rd Annual ACM SIGACT Symposium on Theory of Computing*
[3] Golowich, N., & Livni, R. (2021). 1&. *Advances in Neural Information Processing Systems*

Price of Privacy

VC Dimension characterises PAC learning

Bounds for #mistakes

	Offline Learning (PAC)	Online Learning
Non-private algorithm	$\Theta(\text{VCdim})$	$\Theta(\text{Ldim})$
Private algorithm	$\geq \log^*(\text{Ldim})$ [Alon et al., 2022] $\leq \text{Ldim}^6$ [Ghazi et al., 2021]	$\geq ?$ $\leq O\left(2^{2^{\text{Ldim}}} \frac{1}{\varepsilon} \log T/\delta\right)$ [Golowich & Livni, 2021]

Littlestone Dimension characterises

[1] Alon, N., Bun, M., Livni, R., Malliaris, M., & Moran, S. (2022). Private and online learnability are equivalent. *ACM Journal of the ACM (JACM)*
[2] Ghazi, B., Golowich, N., Kumar, R., & Manurangsi, P. (2021). Sample-efficient proper PAC learning with approximate differential privacy. In *Proceedings of the 53rd Annual ACM SIGACT Symposium on Theory of Computing*
[3] Golowich, N., & Livni, R. (2021). 1&. *Advances in Neural Information Processing Systems*

Objectives

Objectives

1. Design framework to bound privacy loss due to non-private pre-processing for common pre-processors and DP mechanisms

Provable Privacy with Non-Private Pre-Processing

Yaxi Hu *, Amartya Sanyal † and Bernhard Schölkopf‡

Max Planck Institute for Intelligent Systems, Tübingen, Germany

ICML 2024

Objectives

1. Design framework to bound privacy loss due to non-private pre-processing for common pre-processors and DP mechanisms

Provable Privacy with Non-Private Pre-Processing

Yaxi Hu [*], Amartya Sanyal [†] and Bernhard Schölkopf[‡]

Max Planck Institute for Intelligent Systems, Tübingen, Germany

ICML 2024

2. Understand the cost of privacy in DP online learning for the worst online adversary

On the Growth of Mistakes in Differentially Private Online Learning: A Lower Bound Perspective

Daniil Dmitriev[1], Kristóf Szabó[1], and Amartya Sanyal[2]

[1]ETH Zurich
[2]Max Planck Institute for Intelligent Systems, Tübingen

COLT 2024

Non-Private Pre-processing

Renyi DP (Generalisation of (ϵ, δ)-DP)

Renyi DP (Generalisation of (ϵ, δ)-DP)

For $\alpha > 1$, a randomised algorithm $\mathcal{A}$ is $\varepsilon(\alpha)$-RDP if for any two datasets S and S' differing by a single point

$$D_\alpha \left(\mathcal{A}(S) || \mathcal{A}(S')\right) \leq \varepsilon(\alpha)$$

Where D_α denotes α-Rényi divergence between two distributions

Renyi DP (Generalisation of (ϵ, δ)-DP)

For $\alpha > 1$, a randomised algorithm $\mathcal{A}$ is $\varepsilon(\alpha)$-RDP if for any two datasets S and S' differing by a single point

$$D_\alpha \left(\mathcal{A}(S) || \mathcal{A}(S')\right) \leq \varepsilon(\alpha)$$

Privacy loss inequality

Where D_α denotes α-Rényi divergence between two distributions

Renyi DP (Generalisation of (ϵ, δ)-DP)

For $\alpha > 1$, a randomised algorithm $\mathcal{A}$ is $\varepsilon(\alpha)$-RDP if for any two datasets S and S' differing by a single point

$$D_\alpha \left(\mathcal{A}(S) || \mathcal{A}(S')\right) \leq \varepsilon(\alpha)$$

Privacy loss inequality

Where D_α denotes α-Rényi divergence between two distributions

Group RDP

For $\alpha > 1$, if a randomised algorithm $\mathcal{A}$ is $\varepsilon(\alpha)$-RDP, then for any two datasets S and S' differing by m data points,

$$D_\alpha \left(\mathcal{A}(S) || \mathcal{A}(S')\right) \leq m^{1.6} \varepsilon(m\alpha).$$

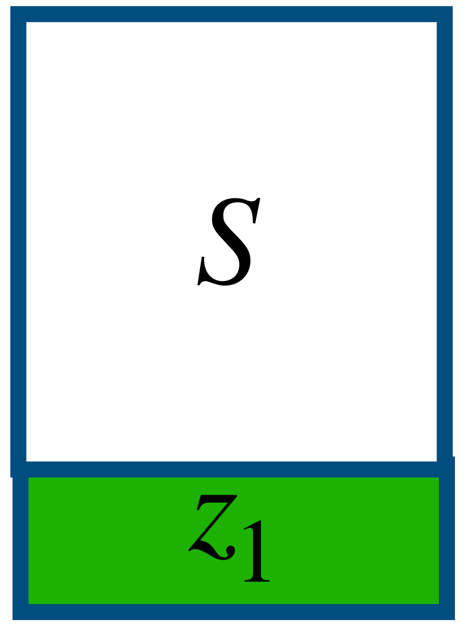
Example of pre-processing: PCA

Example of pre-processing: PCA

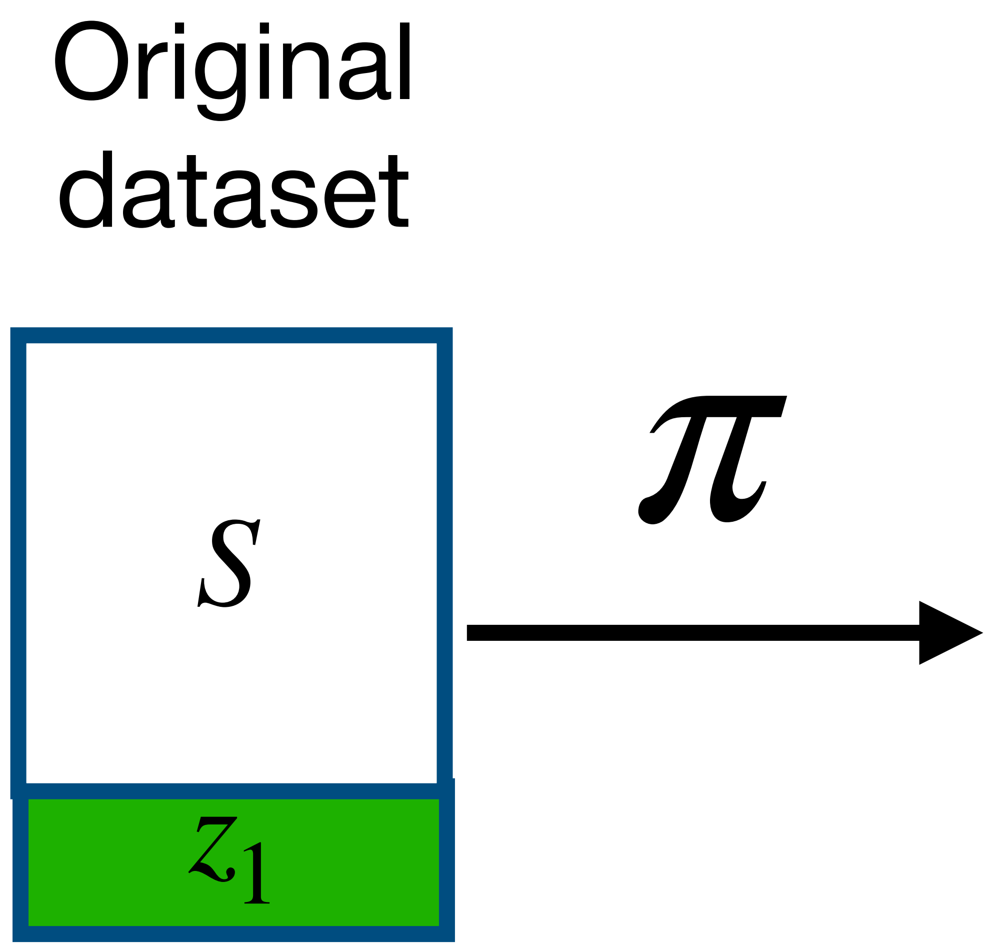
Original
dataset

Example of pre-processing: PCA

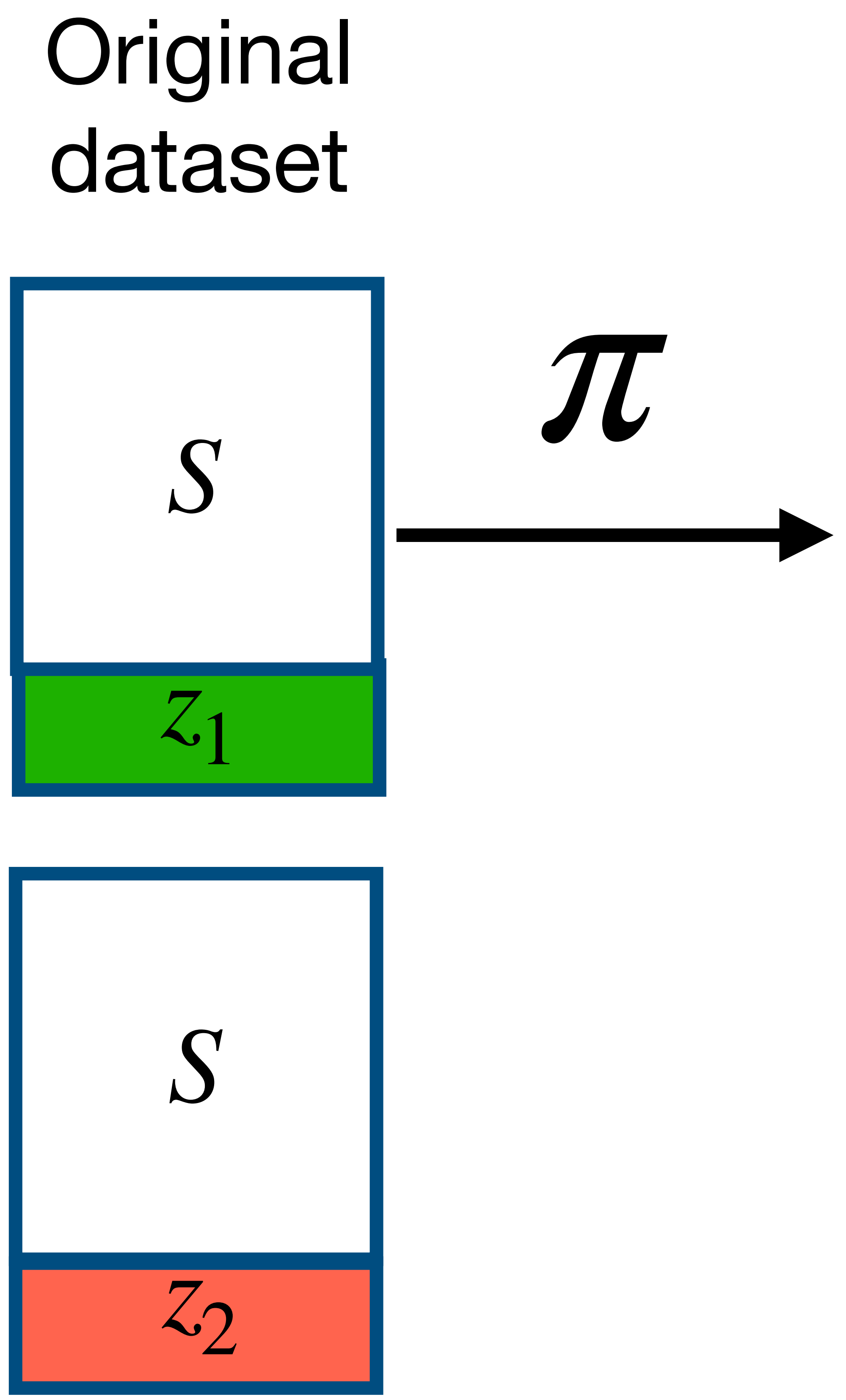
Original dataset



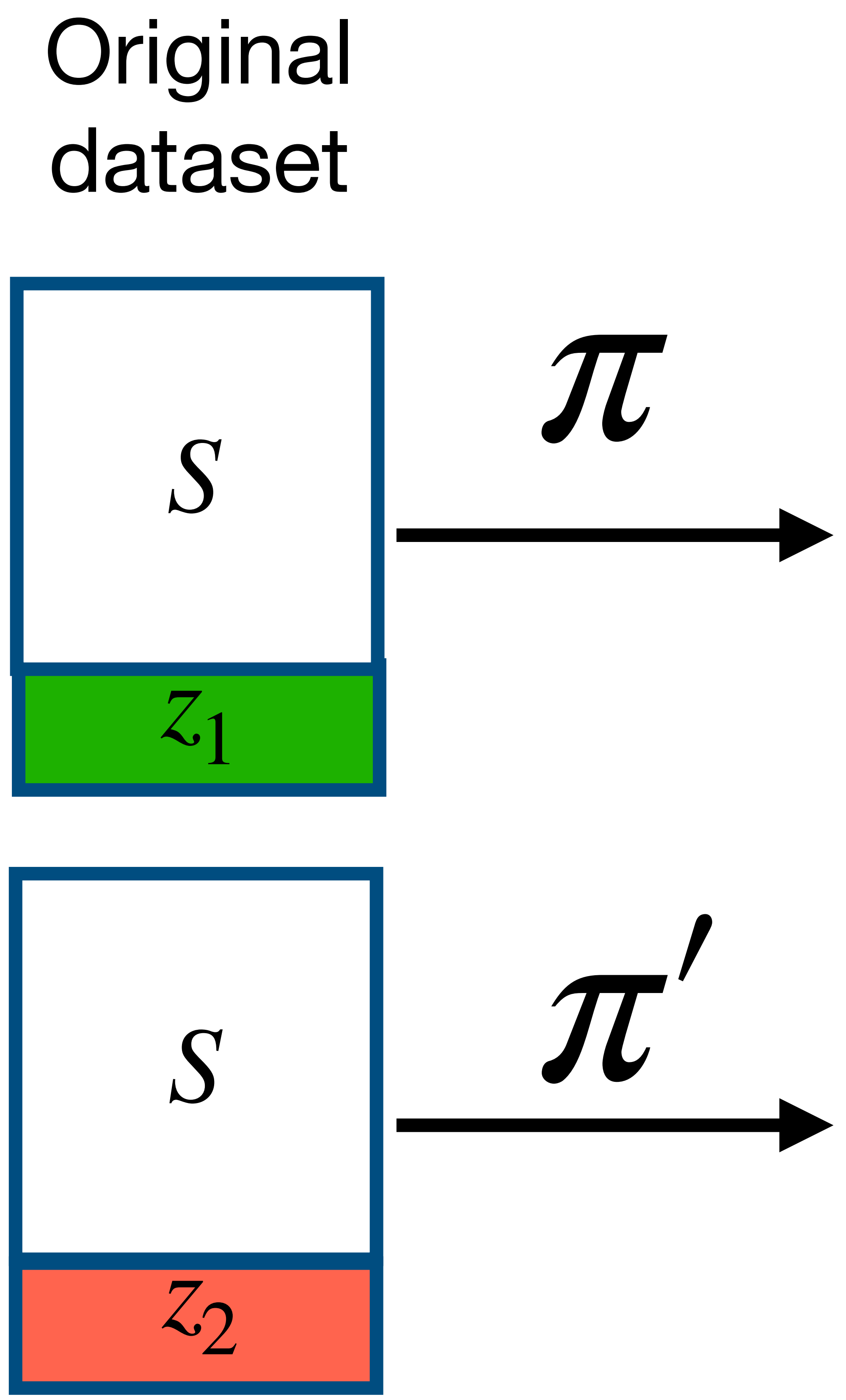
Example of pre-processing: PCA



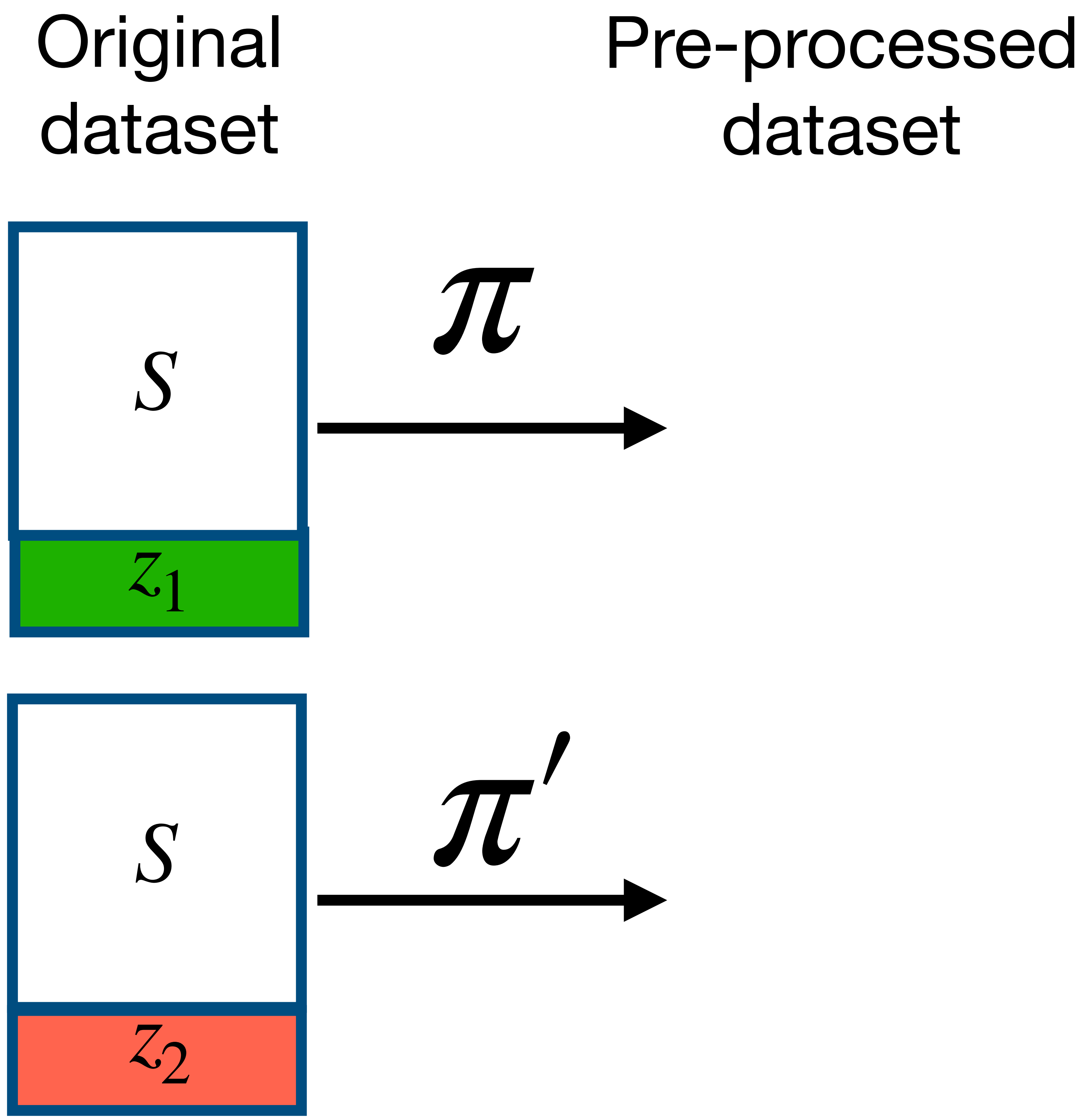
Example of pre-processing: PCA



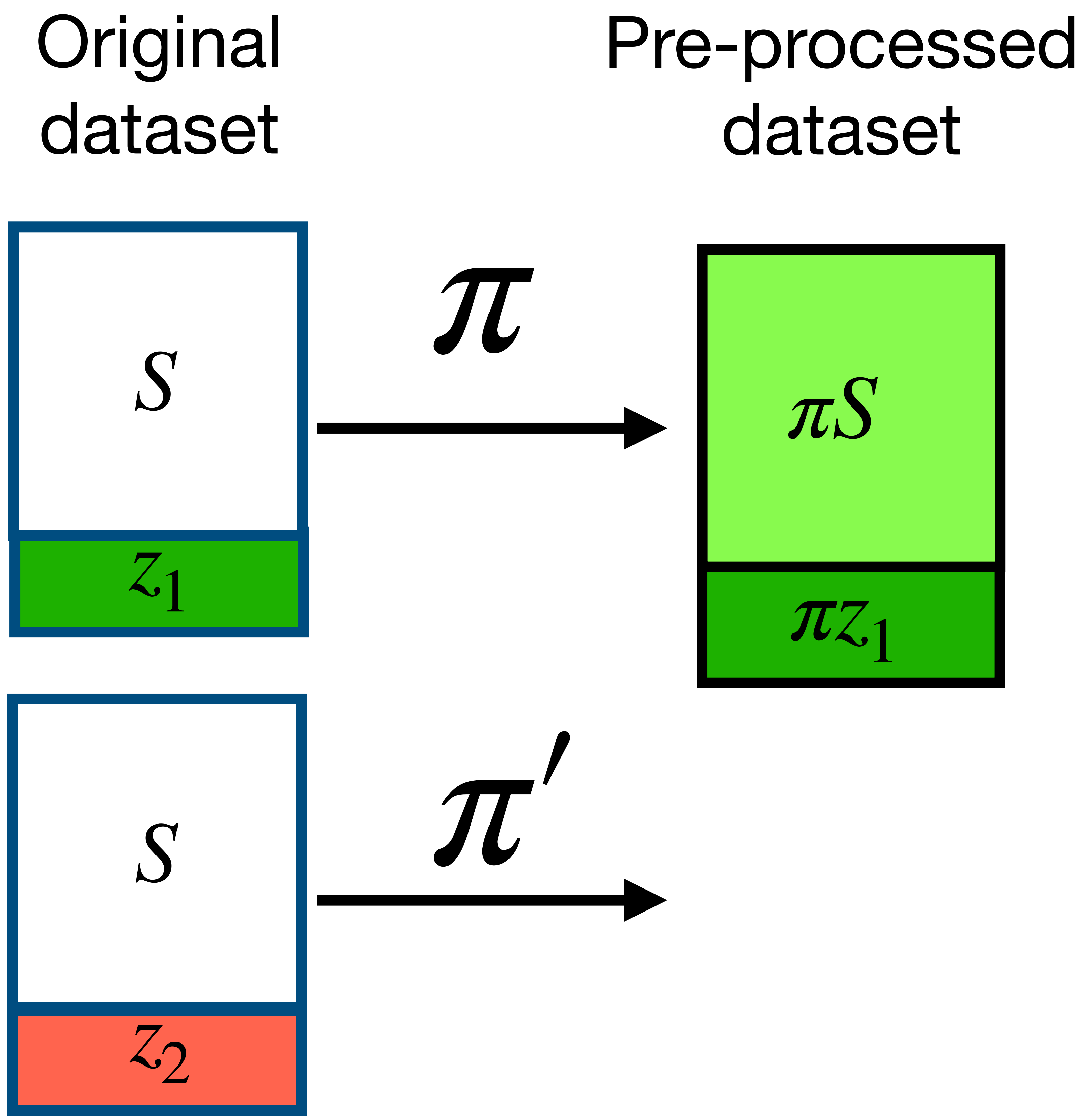
Example of pre-processing: PCA



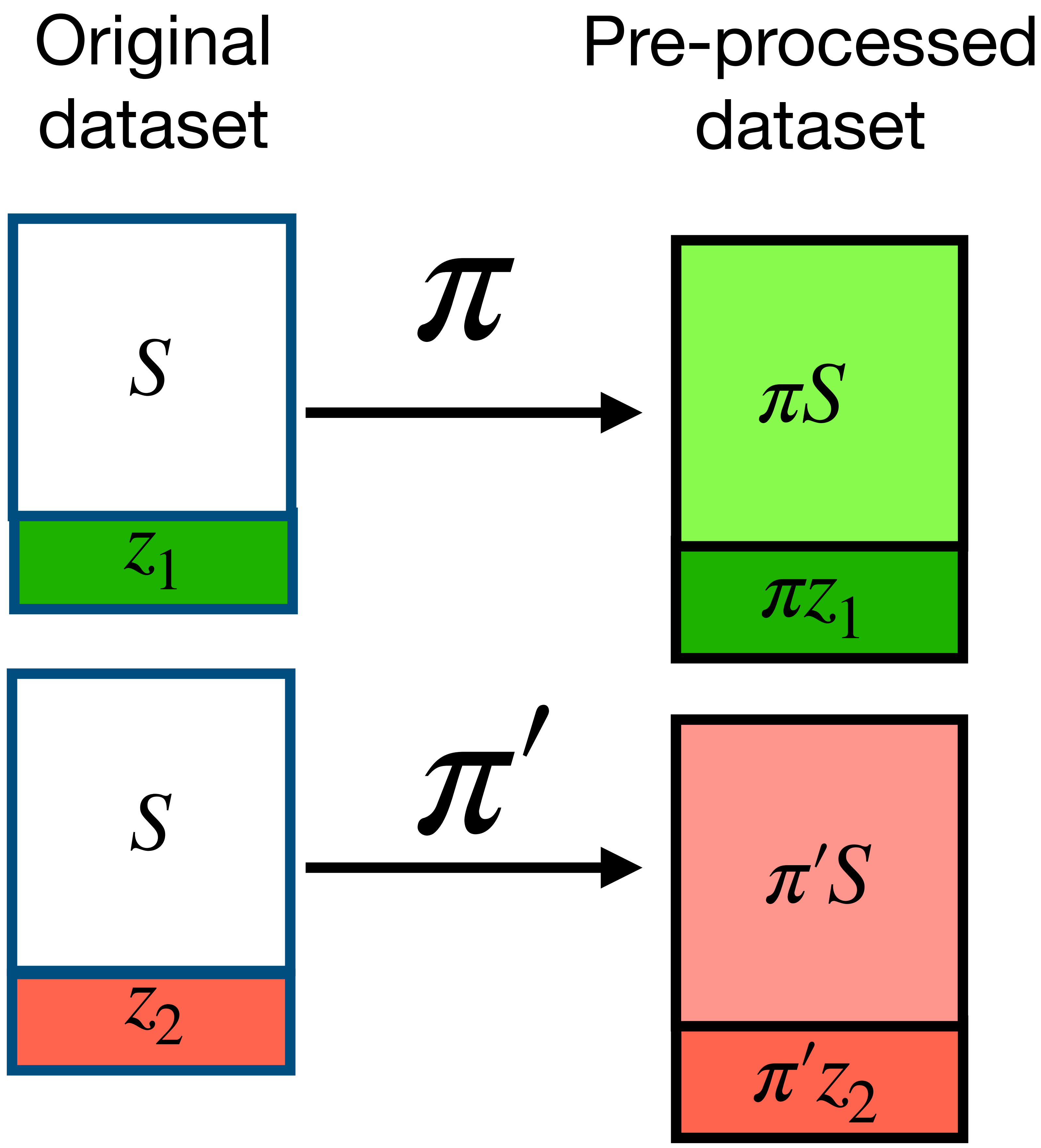
Example of pre-processing: PCA



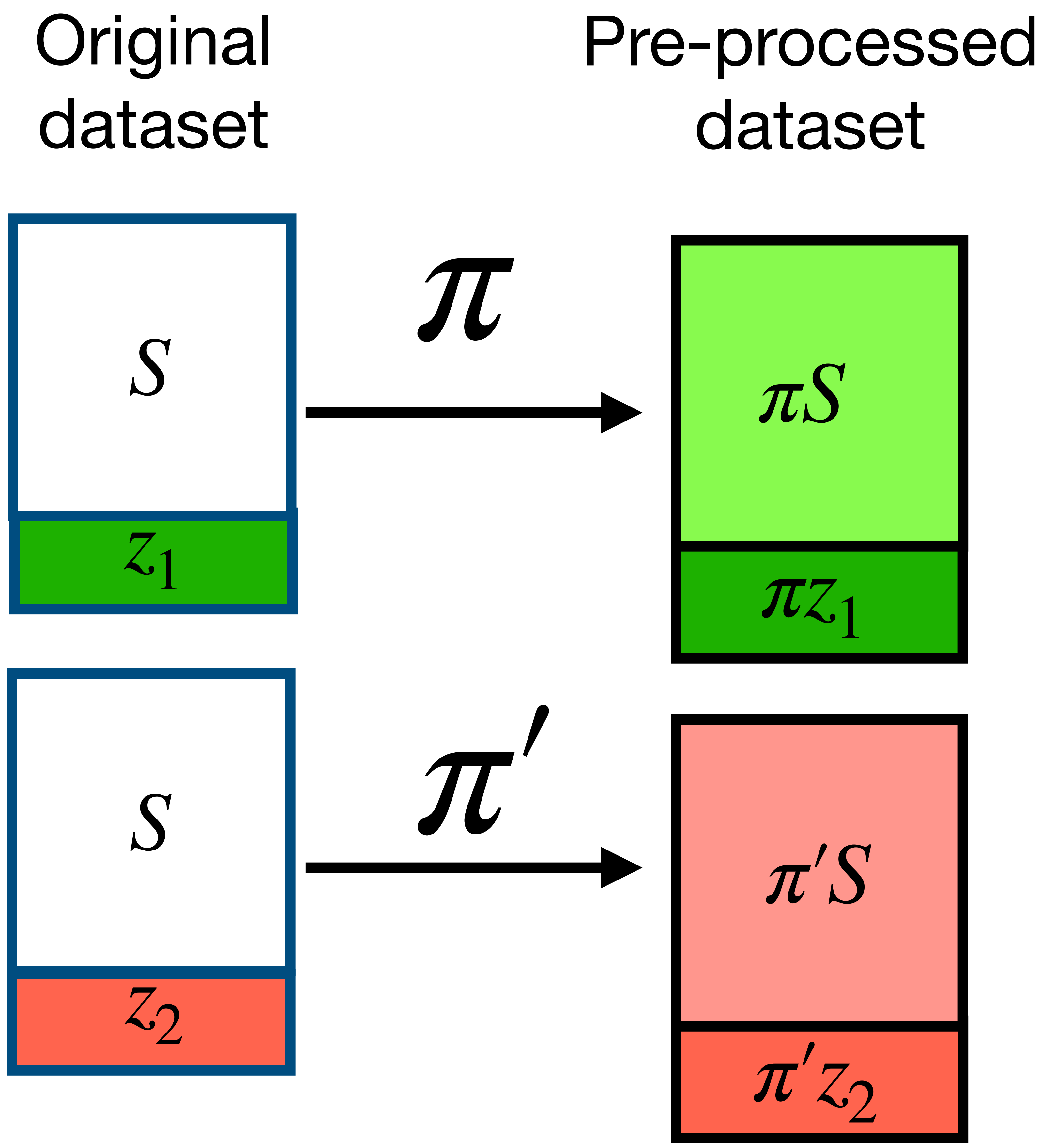
Example of pre-processing: PCA



Example of pre-processing: PCA

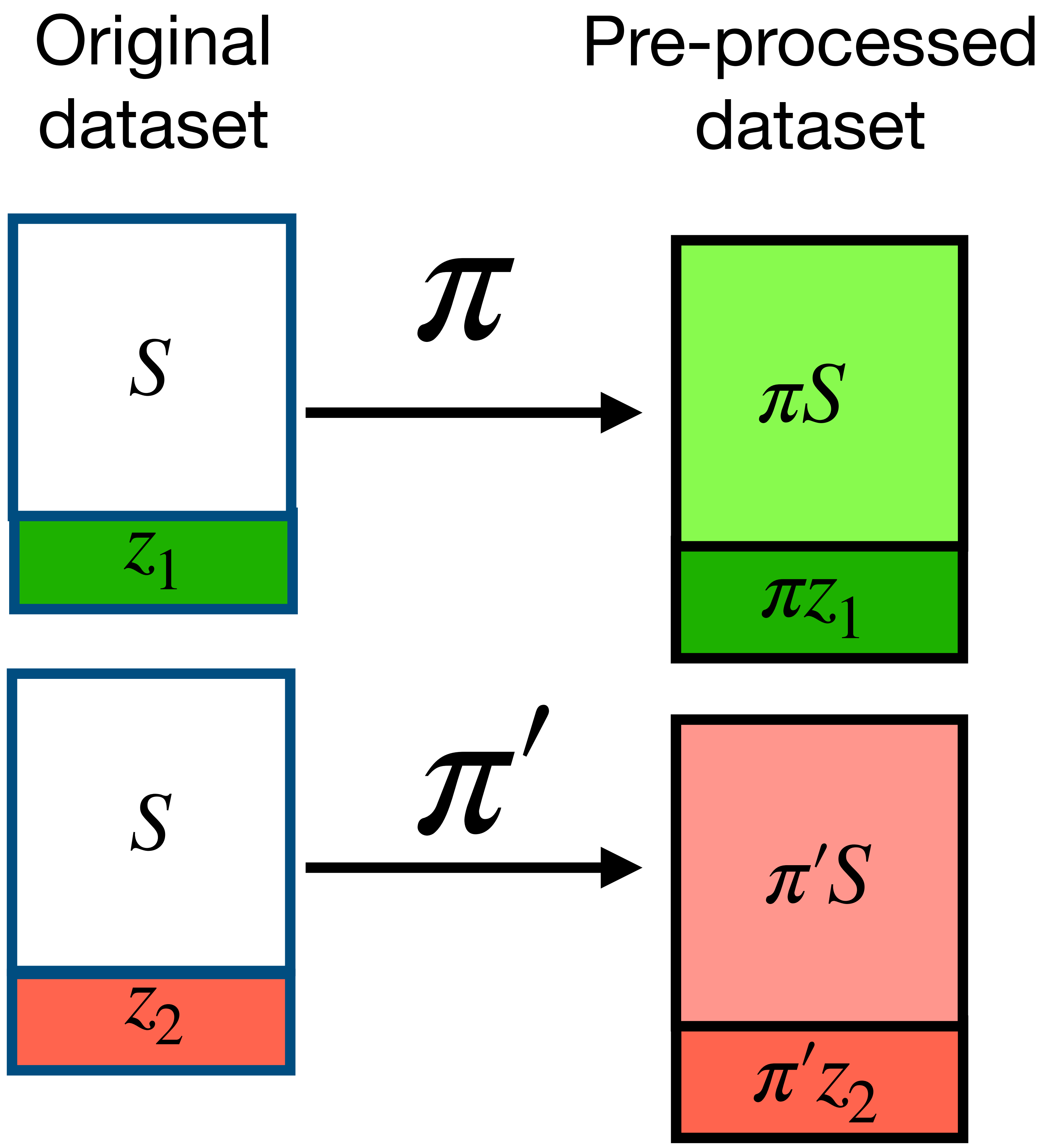


Example of pre-processing: PCA



- Group privacy
  - # different points (i.e. Hamming distance) between pre-processed datasets = n

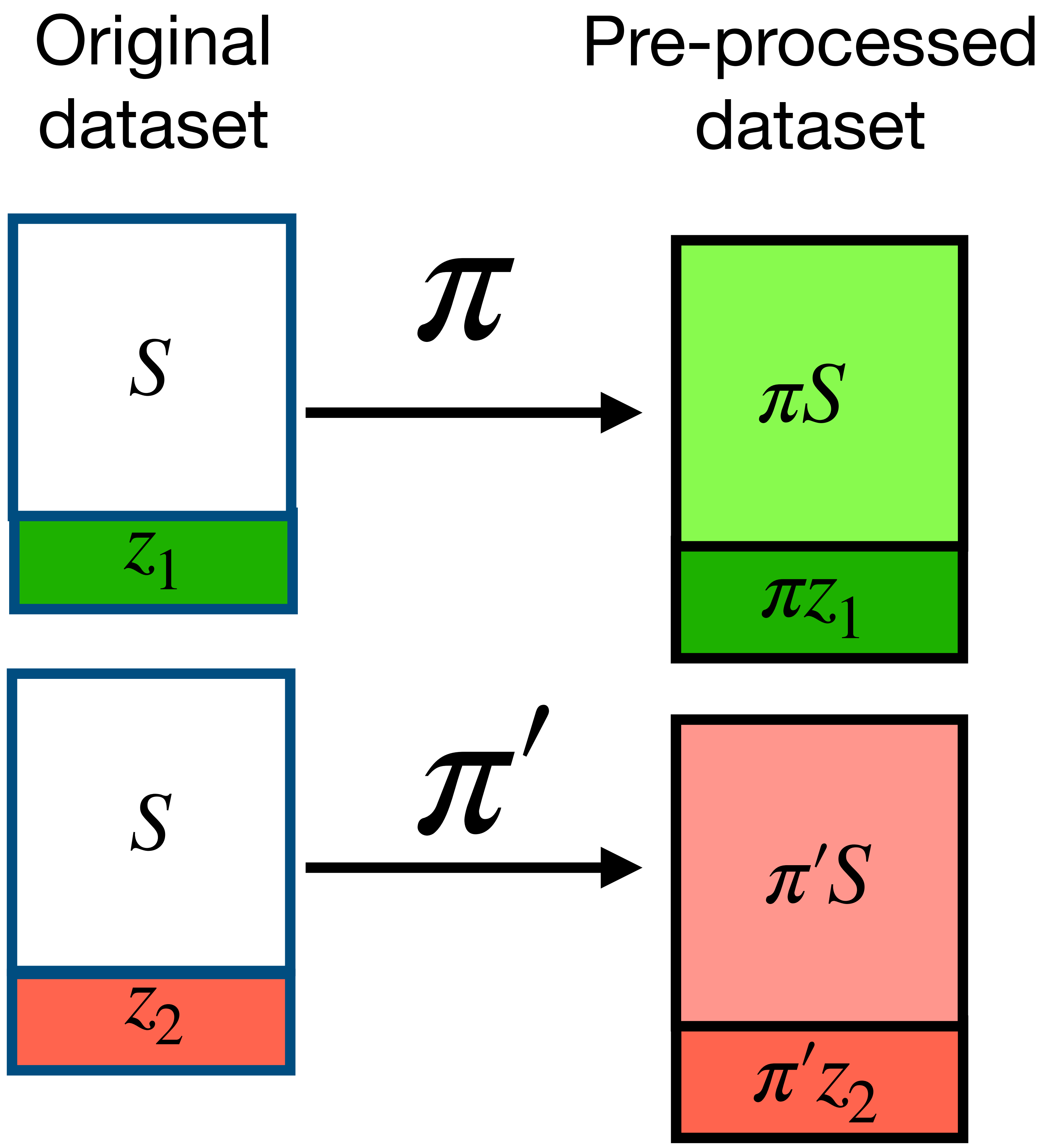
Example of pre-processing: PCA



$$D_\alpha \left(\mathcal{A}(S)||\mathcal{A}(S')\right) \leq n^{1.6}\varepsilon(n\alpha)$$

- Group privacy
  - # different points (i.e. Hamming distance) between pre-processed datasets = n

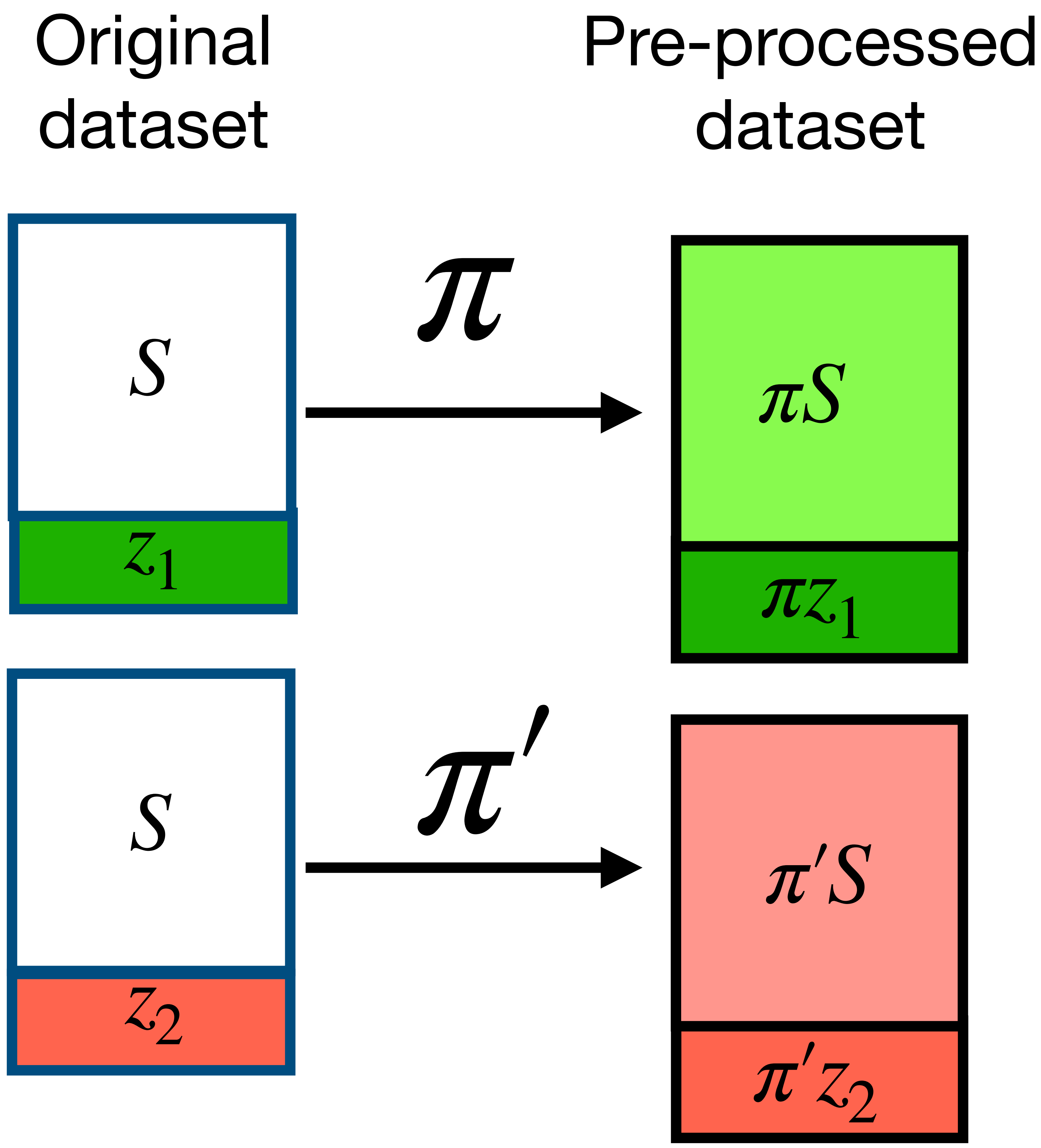
Example of pre-processing: PCA



$$D_\alpha\left(\mathcal{A}(S)||\mathcal{A}(S')\right) \leq n^{1.6}\varepsilon(n\alpha)$$

- Group privacy
  - # different points (i.e. Hamming distance) between pre-processed datasets = n
- Privacy under Euclidean distance?
  - For each point $x \in S$, $\|\pi x - \pi' x\|_2 \leq \|\pi - \pi'\|_2 \|x\|_2 = O(1/n)$
  - Total Distance = $O(1/n)\,.\,n = O(1)$

Example of pre-processing: PCA

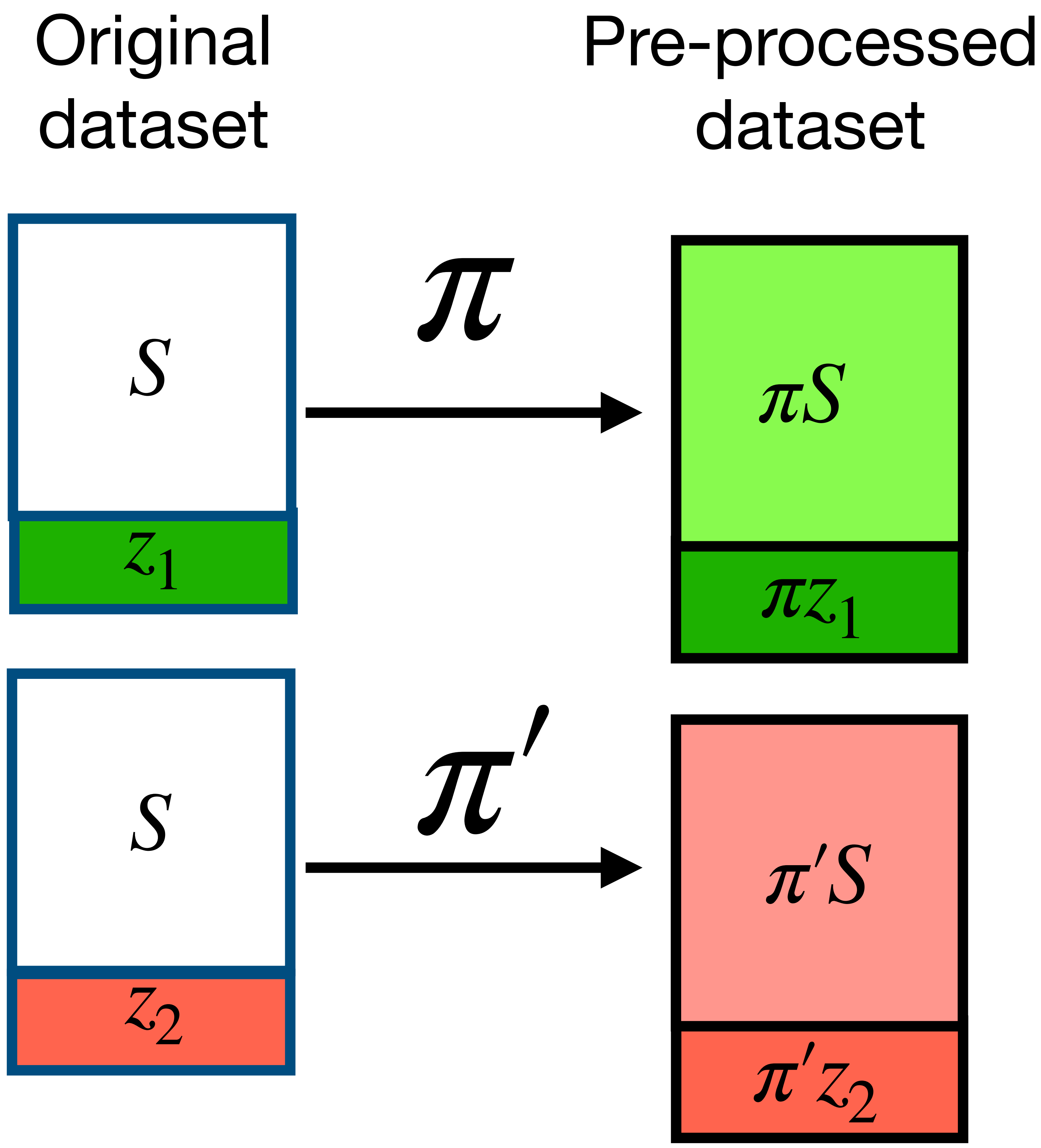


$$D_\alpha\left(\mathcal{A}(S) || \mathcal{A}(S')\right) \leq n^{1.6}\varepsilon(n\alpha)$$

- Group privacy
  - # different points (i.e. Hamming distance) between pre-processed datasets = n
- Privacy under Euclidean distance?
  - For each point $x \in S$, $\|\pi x - \pi' x\|_2 \leq \|\pi - \pi'\|_2 \|x\|_2 = O(1/n)$
  - Total Distance = $O(1/n)\,.\,n = O(1)$

"Privacy defined under a smooth distance metric gives more fine-grained analysis"

Example of pre-processing: PCA



$$D_\alpha\left(\mathcal{A}(S)||\mathcal{A}(S')\right) \leq n^{1.6}\varepsilon(n\alpha)$$

- Group privacy
  - # different points (i.e. Hamming distance) between pre-processed datasets = n
- Privacy under Euclidean distance?
  - For each point $x \in S$, $\|\pi x - \pi' x\|_2 \leq \|\pi - \pi'\|_2 \|x\|_2 = O(1/n)$
  - Total Distance = $O(1/n)\,.\,n = O(1)$

“Privacy defined under a sr … rained analysis”

Smooth RDP

Renyi DP and Smooth Renyi DP (SRDP)

Renyi DP and Smooth Renyi DP (SRDP)

For $\alpha > 1, m \in \mathbb{N}$, an algorithm $\mathscr{A}$ is $\varepsilon(\alpha)$-RDP if for any two datasets S and S' differing at m points- $d_H(S, S') \leq m$

$$D_\alpha\left(\mathcal{A}(S)||\mathcal{A}(S')\right) \leq m^{1.6}\varepsilon(m\alpha)$$

Renyi DP and Smooth Renyi DP (SRDP)

For $\alpha > 1, m \in \mathbb{N}$, an algorithm $\mathscr{A}$ is $\varepsilon(\alpha)$-RDP if for any two datasets S and S' differing at m points- $d_H(S, S') \leq m$

$$D_\alpha\left(\mathcal{A}(S) || \mathcal{A}(S')\right) \leq m^{1.6} \varepsilon(m\alpha)$$

Stability against arbitrary perturbation on a fixed number of data points

Renyi DP and Smooth Renyi DP (SRDP)

For $\alpha > 1, m \in \mathbb{N}$, an algorithm $\mathscr{A}$ is $\varepsilon(\alpha)$-RDP if for any two datasets S and S' differing at m points- $d_H(S, S') \leq m$

$$D_\alpha\left(\mathcal{A}(S) || \mathcal{A}(S')\right) \leq m^{1.6} \varepsilon(m\alpha)$$

Stability against arbitrary perturbation on a fixed number of data points

For $\alpha > 1, \tau > 0$, an algorithm $\mathcal{A}$ is $\varepsilon(\alpha, \tau)$-SRDP if for any two datasets S and S' satisfying

$$\sum_{i=1}^{n} \|S_i - S_i'\|_2 \leq \tau$$

$$D_\alpha\left(\mathcal{A}(S) || \mathcal{A}(S')\right) \leq \varepsilon\left(\alpha, \tau\right)$$

Renyi DP and Smooth Renyi DP (SRDP)

For $\alpha > 1, m \in \mathbb{N}$, an algorithm $\mathscr{A}$ is $\varepsilon(\alpha)$-RDP if for any two datasets S and S' differing at m points-

$d_H(S, S') \leq m$

$$D_\alpha\left(\mathcal{A}(S) || \mathcal{A}(S')\right) \leq m^{1.6} \varepsilon(m\alpha)$$

Stability against arbitrary perturbation on a fixed number of data points

For $\alpha > 1, \tau > 0$, an algorithm $\mathcal{A}$ is $\varepsilon(\alpha, \tau)$-SRDP if for any two datasets S and S' satisfying

$$\sum_{i=1}^{n} \|S_i - S_i'\|_2 \leq \tau$$

$$D_\alpha\left(\mathcal{A}(S) || \mathcal{A}(S')\right) \leq \varepsilon\left(\alpha, \tau\right)$$

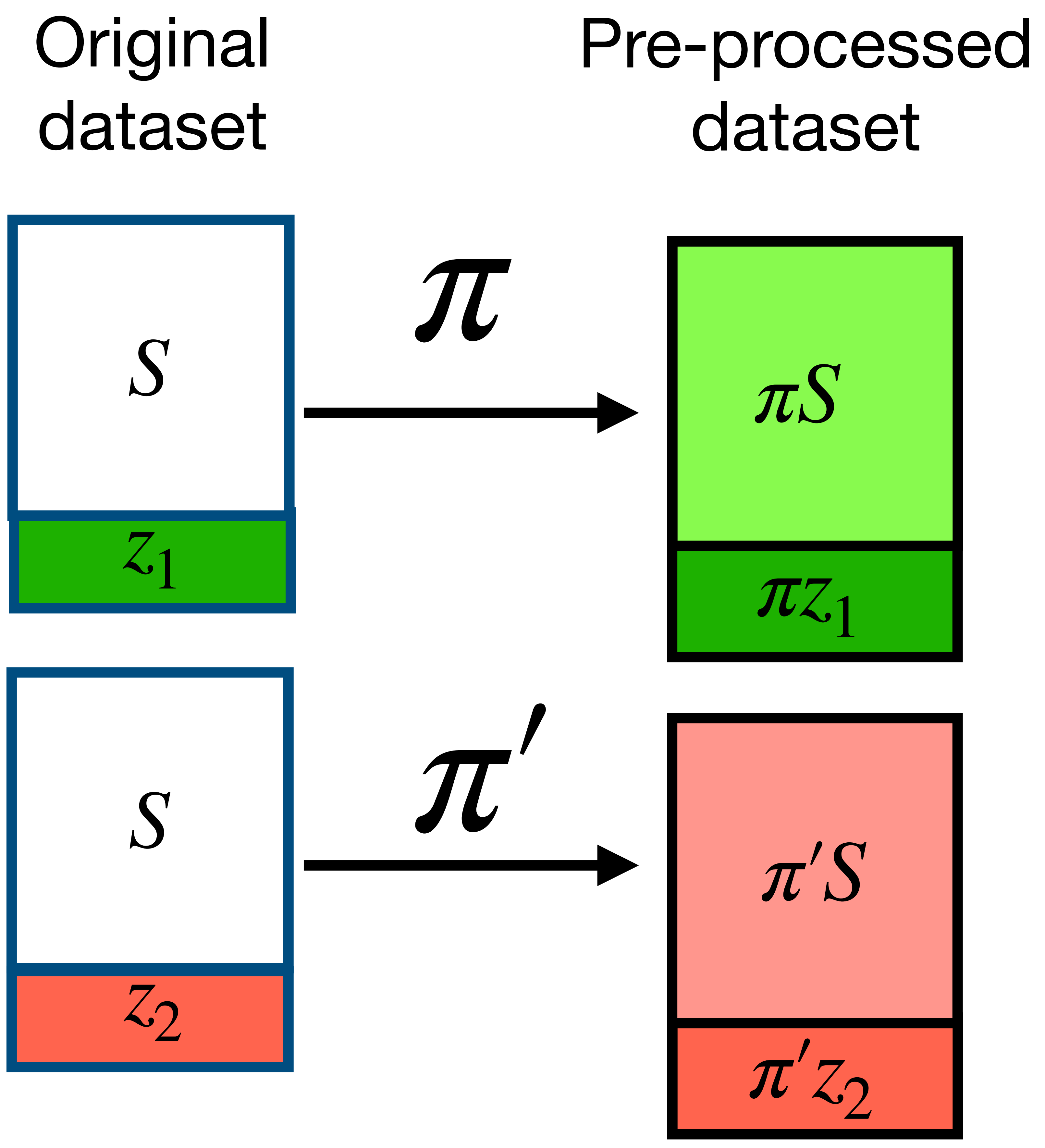
Stability against bounded perturbation on arbitrary number of data points

Most DP mechanisms satisfy Smooth-RDP

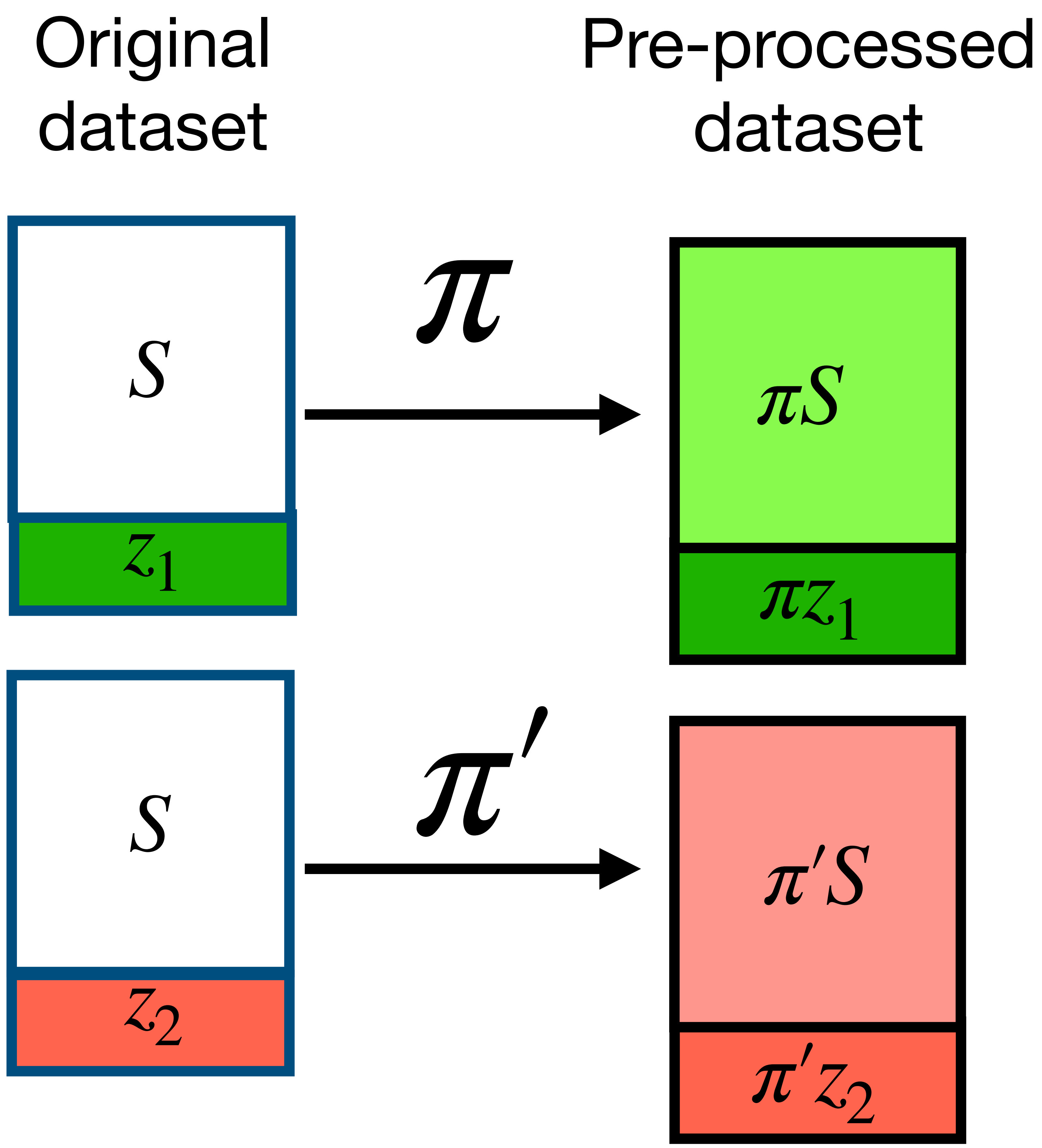
Notation	Meaning	Mechanism	Assumptions	RDP	SRDP
f	Output function	$\mathcal{M}_G$	f is L-Lipschitz	$\frac{\alpha\varepsilon^2}{2}$	$\frac{\alpha L^2\tau^2\varepsilon^2}{2\Delta_f^2}$
		$\mathcal{M}_L$	f is L-Lipschitz	ε	$\frac{L\tau}{\Delta_f}\varepsilon$
Q	Score function	$\mathcal{M}_E$	Q is L-Lipschitz	ε	$\frac{L\tau}{\Delta_Q}\varepsilon$
ℓ	Loss function	$\mathcal{A}_{\text{GD}}$	ℓ is L-Lipschitz and μ-smooth, $\sigma = \frac{L\sqrt{T}}{\varepsilon n}$	$2\alpha\varepsilon^2$	$\frac{\alpha\mu^2\tau^2\varepsilon^2}{2L^2}$
T	Number of iteration	$\mathcal{A}_{\text{SGD}-\text{samp}}$	ℓ is L-Lipschitz and μ-smooth, $\sigma = \Omega(L\sqrt{T}/\varepsilon n)$, inverse point-wise divergence γ, $1 \leq \alpha \leq \min\left\{\frac{\sqrt{T}}{\varepsilon}, \frac{L^2T}{\varepsilon^2 n^2}\log\frac{n^2\epsilon}{L\sqrt{T}}\right\}$	$\frac{\alpha^2\varepsilon^2}{2}$	$\frac{\alpha\mu^2\tau^2\varepsilon^2\gamma^2}{2L^2}$
η	Learning rate	$\mathcal{A}_{\text{SGD}-\text{iter}}$	ℓ is convex, L-Lipschitz and μ-smooth, $\sigma = \frac{8\sqrt{2\log n}\eta L}{\varepsilon\sqrt{n}}$, $\varepsilon = O(1/n\alpha^2)$, maximum divergence κ_τ, $L\sqrt{2\alpha(\alpha-1)} \leq \sigma$	$\frac{\alpha\varepsilon^2}{2}$	$\frac{\alpha\tau^2\mu^2 n\log(n-\kappa_\tau+2)}{2(n-\kappa_\tau+1)L^2\log n}$

Table 1: RDP and SRDP parameters of DP mechanisms.

Sensitivity of pre-processing algorithms

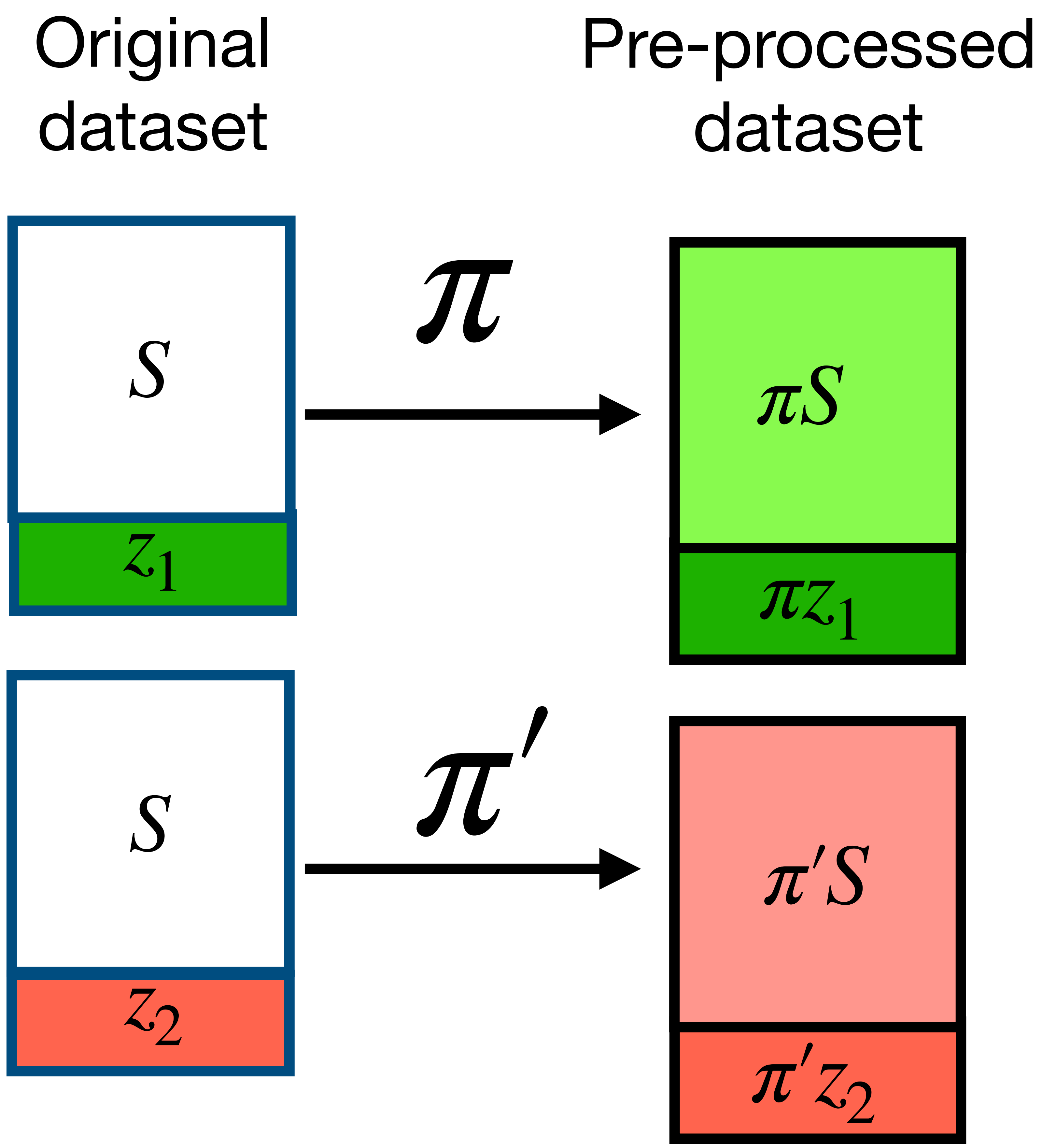


Sensitivity of pre-processing algorithms



$$L_2 \text{ sensitivity} = \max_{x \in S} \max_{S,S'} \|\pi_S(x) - \pi_{S'}(x)\|_2$$

Sensitivity of pre-processing algorithms

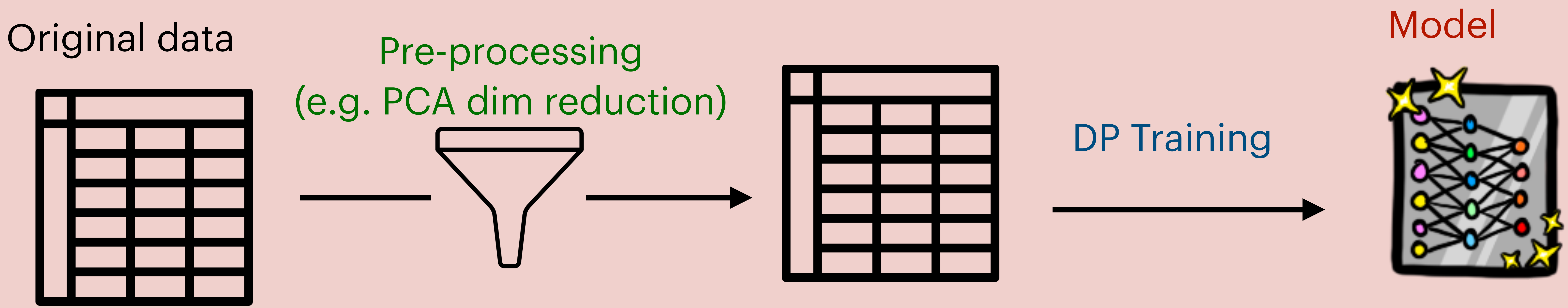


$$L_2 \text{ sensitivity} = \max_{x \in S} \max_{S,S'} \|\pi_S(x) - \pi_{S'}(x)\|_2$$

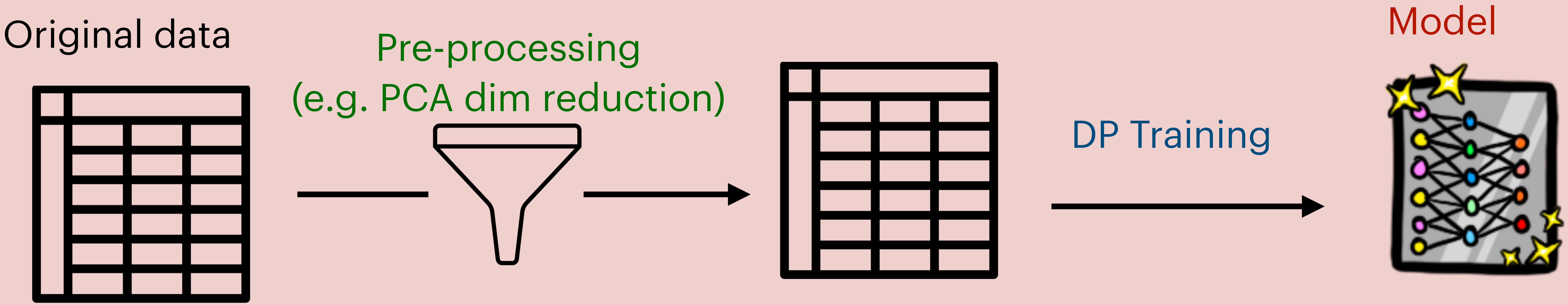
$$L_\infty \text{ sensitivity: } \max_{x \in S} \max_{S,S'} \|\pi_S(x) - \pi_{S'}(x)\|_0$$

Main Result

Main Result

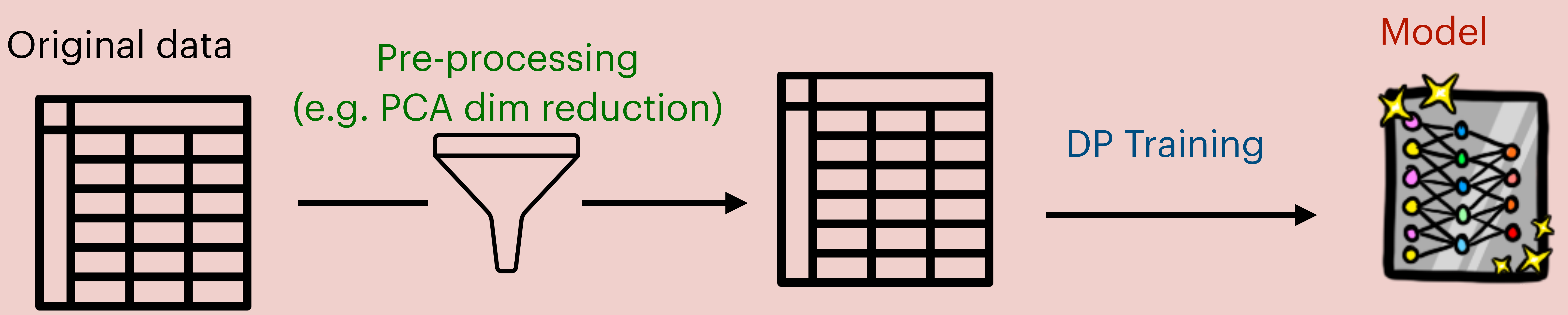

Original data
Pre-processing
(e.g. PCA dim reduction)
DP Training
Model

Main Result



Assumption 1:
The pre-processing algorithm has
L_2 sensitivity Δ_2 and
L_∞ sensitivity Δ_∞

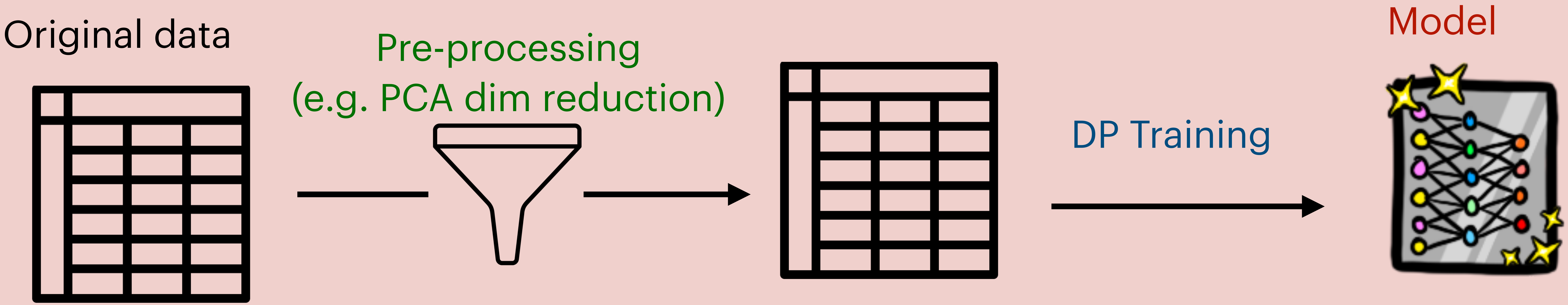
Main Result



Assumption 1:
The pre-processing algorithm has
L_2 sensitivity Δ_2 and
L_∞ sensitivity Δ_∞

Assumption 2:
The training algorithm is
$(\alpha, \varepsilon(\alpha))$-RDP and
$(\alpha, \tilde{\varepsilon}(\alpha, \tau))$-SRDP

Main Result



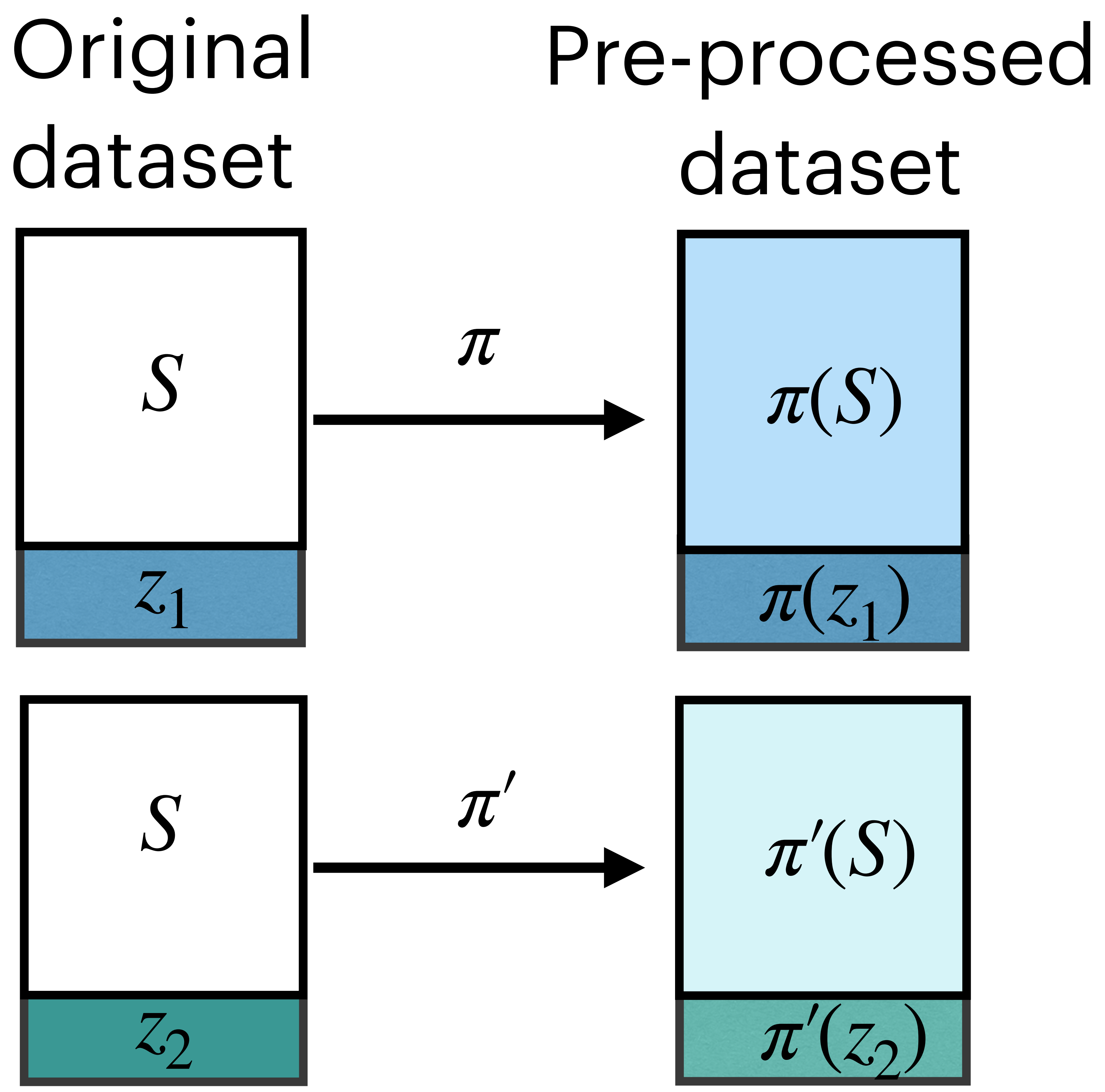
Assumption 1:
The pre-processing algorithm has
L_2 sensitivity Δ_2 and
L_∞ sensitivity Δ_∞

Assumption 2:
The training algorithm is
$(\alpha, \varepsilon(\alpha))$-RDP and
$(\alpha, \tilde{\varepsilon}(\alpha, \tau))$-SRDP

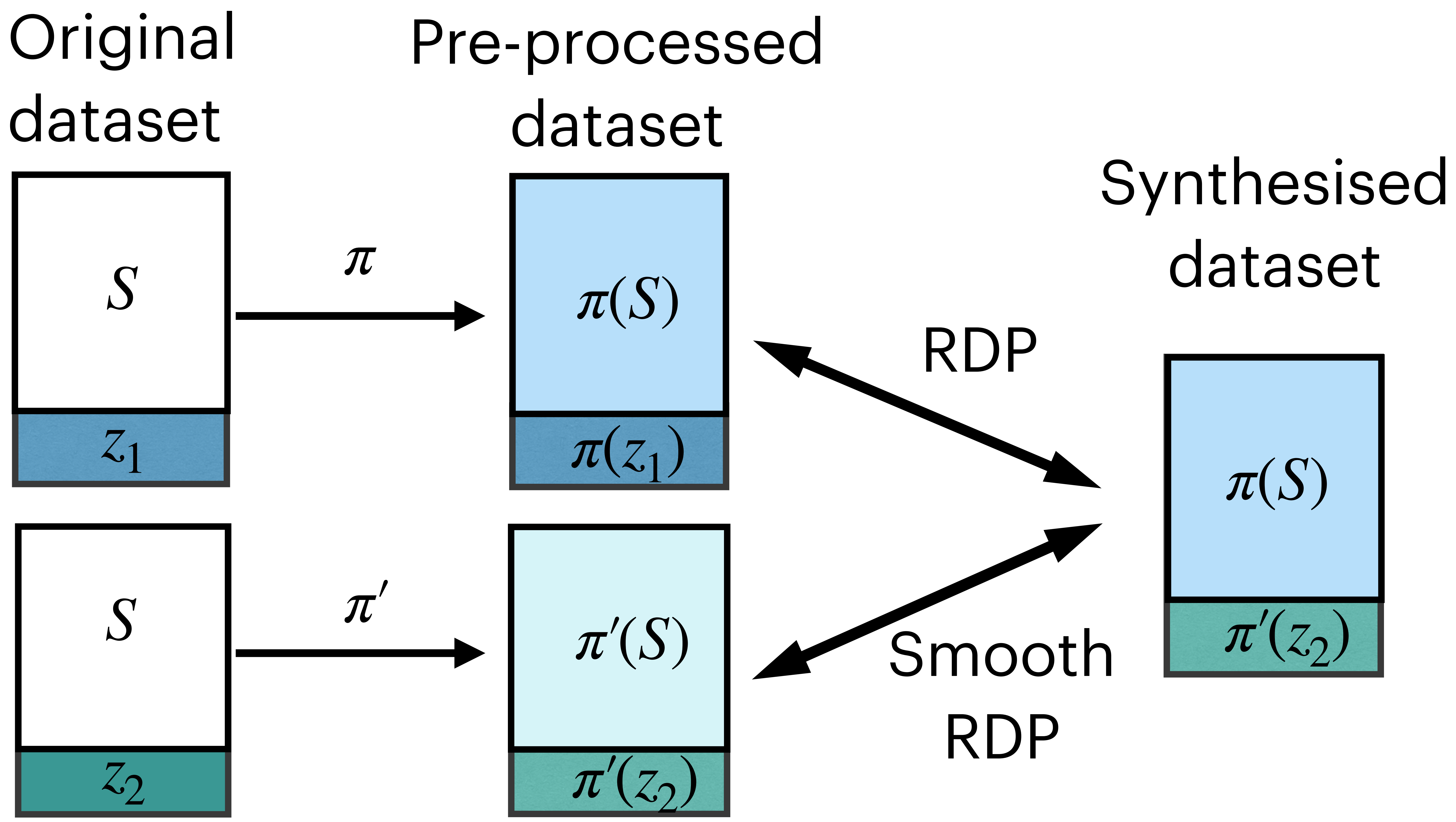
The pre-processed DP pipeline is $(\alpha, \hat{\varepsilon})$-RDP, where

$$\hat{\varepsilon} = \frac{2\alpha - 1}{2(\alpha - 1)}\left(\tilde{\varepsilon}(2\alpha, \Delta_2\Delta_\infty) + \varepsilon(2\alpha)\right).$$

Proof Sketch

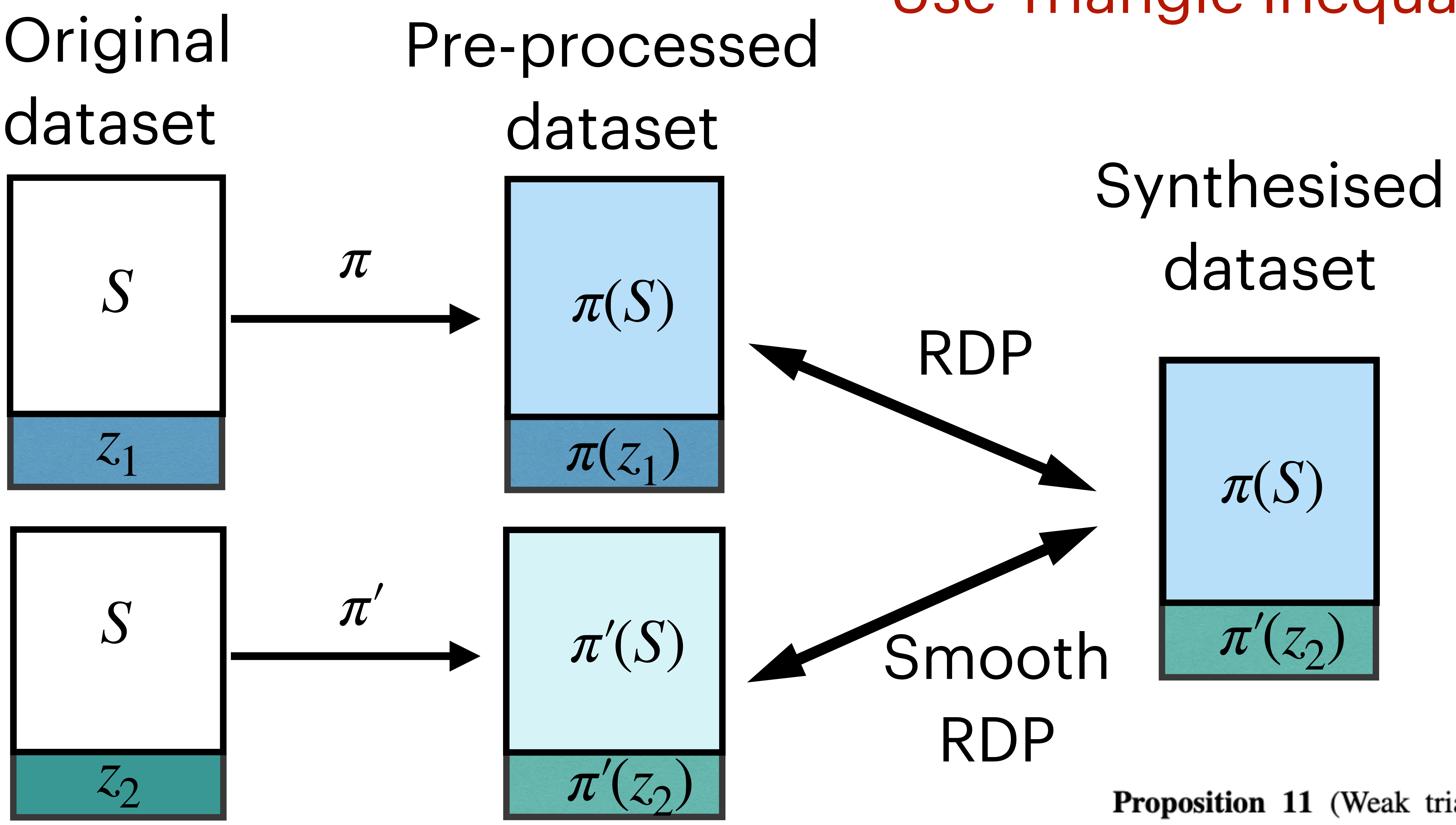


Proof Sketch



Proof Sketch

Use Triangle Inequality for Renyi DP



Proposition 11 (Weak triangle inequality). *Let P, Q, R be distributions on $\mathcal{R}$. Then for $\alpha > 1$ and for any $p, q > 1$ satisfying $1/p + 1/q = 1$ it holds that*

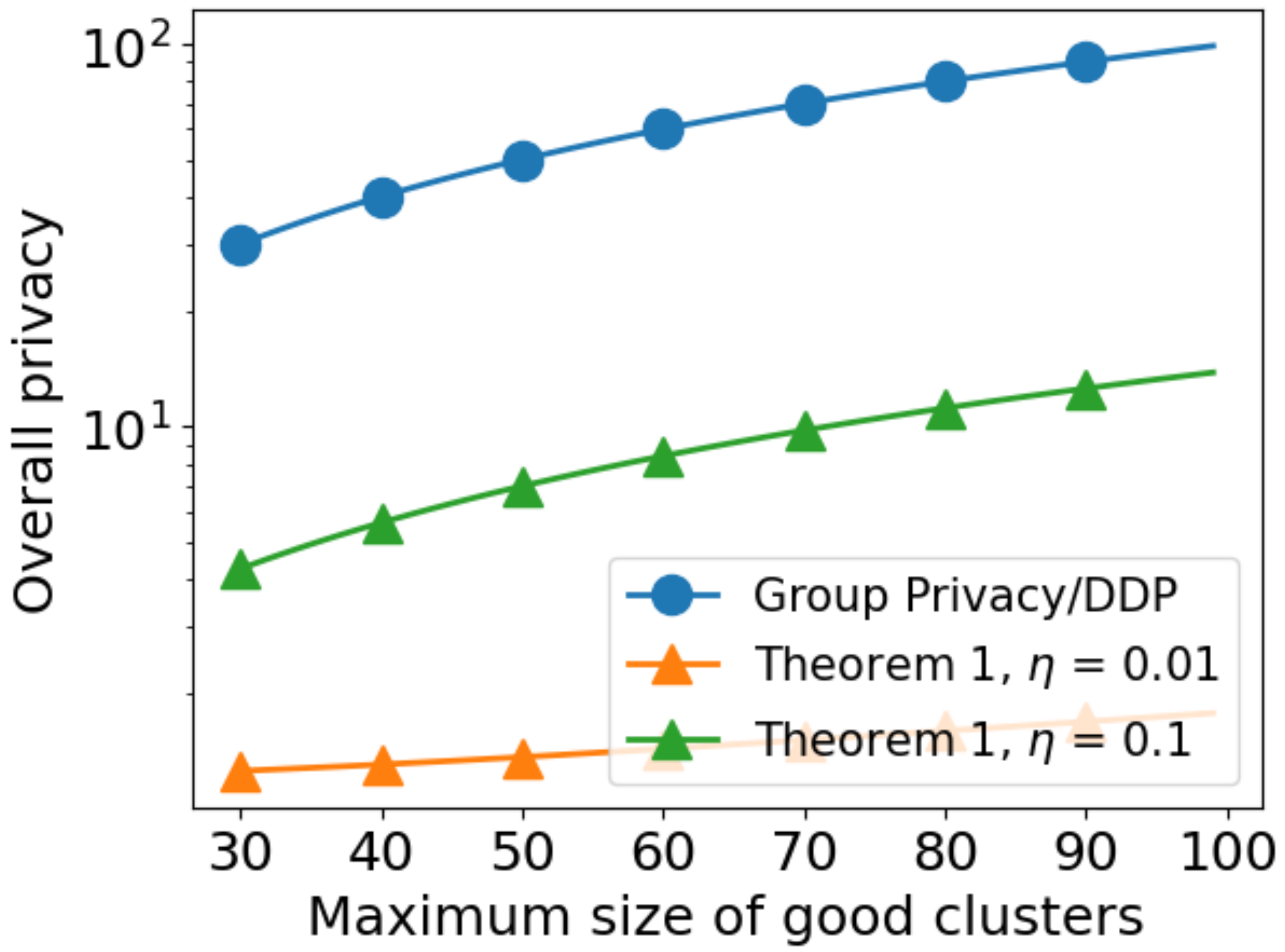
$$D_\alpha(P\|Q) \le \frac{\alpha - 1/p}{\alpha - 1} D_{p\alpha}(P\|R) + D_{q(\alpha-1/p)}(R\|Q).$$

Visualising the advantage

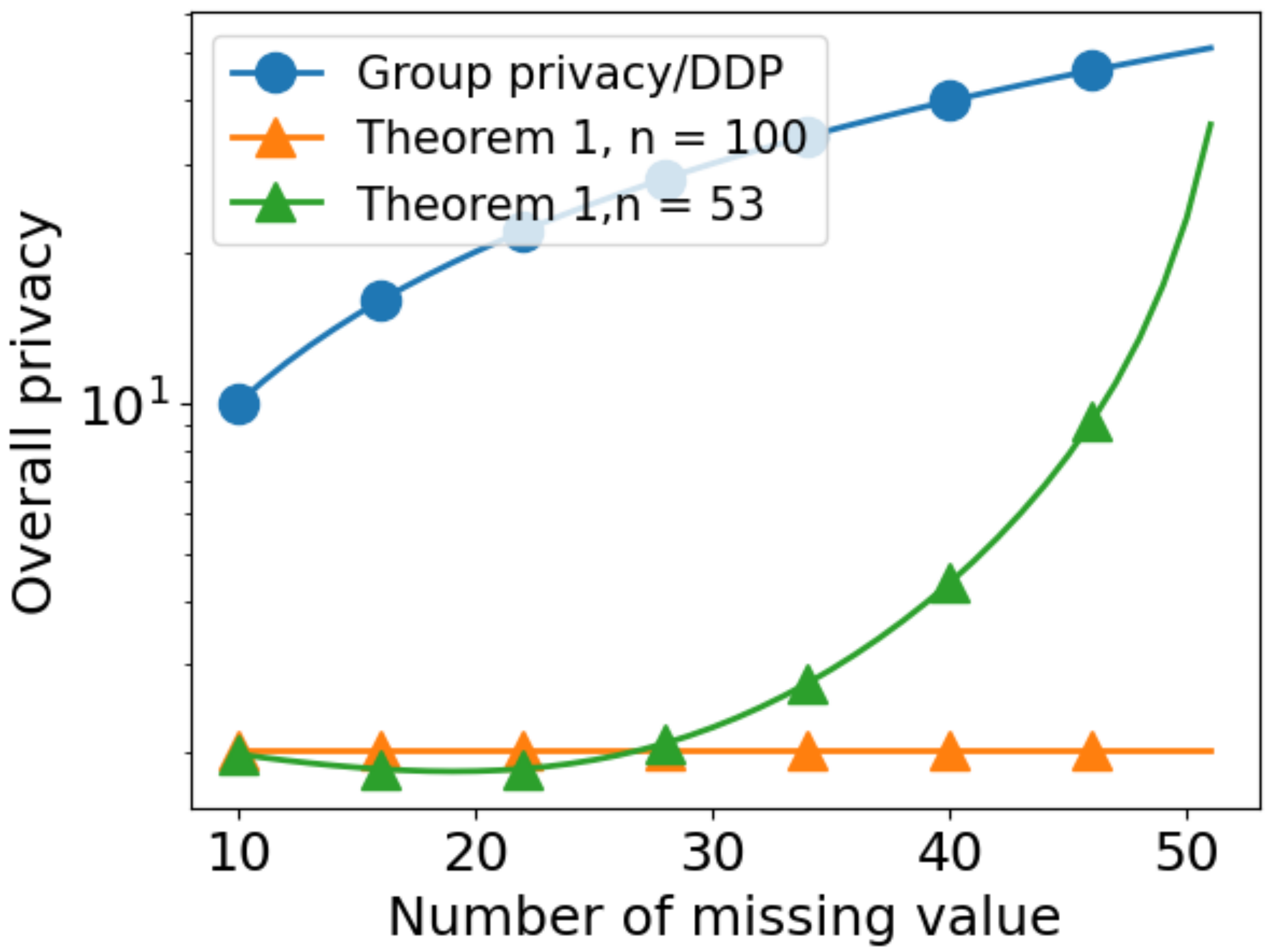
Visualising the advantage

	Quantization	Mean imputation	PCA	Standard Scaling
$\mathcal{M}_G$	$1.05\alpha\varepsilon^2\left(1+\eta^2p^2\right)$	$1.05\alpha\varepsilon^2\left(1+\frac{4p^2}{(n-p)^2}\right)$	$1.05\alpha\varepsilon^2\left(1+\frac{12.2^2}{(\delta^k_{\min})^2}\right)$	$1.05\alpha\varepsilon^2\left(1+\frac{4}{\sigma^3_{\min}}\right)$
$\mathcal{A}_{\mathrm{GD}}$	$1.05\alpha\varepsilon^2\left(4+\eta^2p^2\right)$	$4.2\alpha\varepsilon^2\left(1+\frac{p^2}{(n-p)^2}\right)$	$1.05\alpha\varepsilon^2\left(4+\frac{12.2^2}{(\delta^k_{\min})^2}\right)$	$4.2\alpha\varepsilon^2\left(1+\frac{1}{\sigma^3_{\min}}\right)$
$\mathcal{M}_L/\mathcal{M}_E$	$\varepsilon\left(1+\eta p\right)$	$\varepsilon\left(1+\frac{2p}{n-p}\right)$	$\varepsilon\left(1+\frac{12.2}{\delta^k_{\min}}\right)$	$\varepsilon\left(1+\frac{4}{\sigma^3_{\min}}\right)$
$\mathcal{A}_{\mathrm{SGD-samp}}$	–	–	$1.05\alpha\varepsilon^2\left(2\alpha+\frac{12.2^2}{(\delta^k_{\min})^2}\right)$	$2.1\alpha\varepsilon^2\left(\alpha+\frac{8}{\sigma^6_{\min}}\right)$
$\mathcal{A}_{\mathrm{SGD-iter}}$	$1.1\alpha\varepsilon^2\left(1+\frac{\eta^2p^2\frac{n}{\log n}}{\frac{n-p}{\log n-p}}\right)$	$1.1\alpha\varepsilon^2\left(1+\frac{4p^2\frac{n}{\log n}}{\frac{(n-p)^3}{\log n-p}}\right)$	–	–

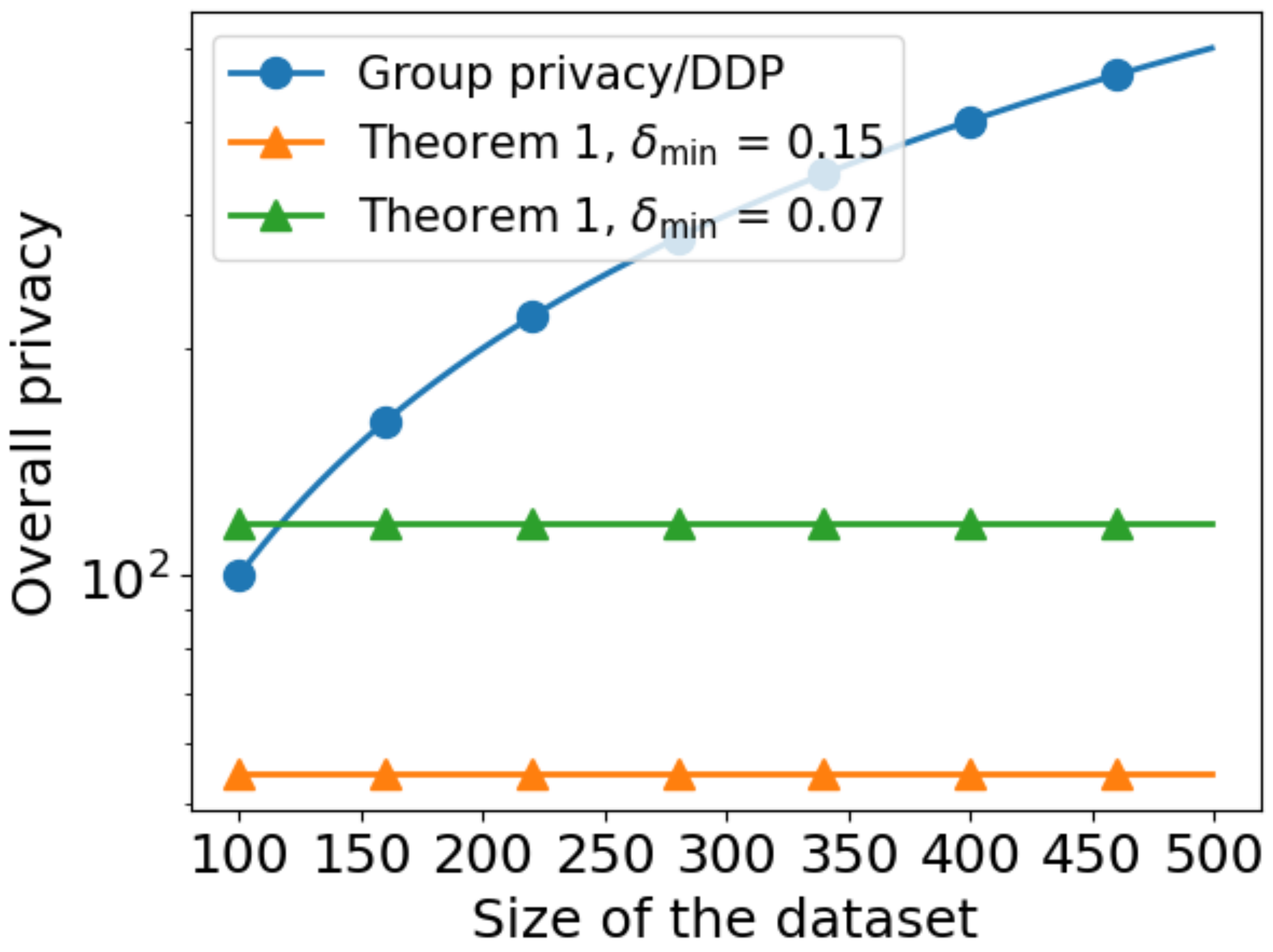
Visualising the advantage



Quantization



Mean imputation



PCA

	Quantization	Mean imputation	PCA	Standard Scaling
$\mathcal{M}_G$	$1.05\alpha\varepsilon^2\left(1+\eta^2p^2\right)$	$1.05\alpha\varepsilon^2\left(1+\frac{4p^2}{(n-p)^2}\right)$	$1.05\alpha\varepsilon^2\left(1+\frac{12.2^2}{(\delta^k_{\min})^2}\right)$	$1.05\alpha\varepsilon^2\left(1+\frac{4}{\sigma^3_{\min}}\right)$
$\mathcal{A}_{\mathrm{GD}}$	$1.05\alpha\varepsilon^2\left(4+\eta^2p^2\right)$	$4.2\alpha\varepsilon^2\left(1+\frac{p^2}{(n-p)^2}\right)$	$1.05\alpha\varepsilon^2\left(4+\frac{12.2^2}{(\delta^k_{\min})^2}\right)$	$4.2\alpha\varepsilon^2\left(1+\frac{1}{\sigma^3_{\min}}\right)$
$\mathcal{M}_L/\mathcal{M}_E$	$\varepsilon\left(1+\eta p\right)$	$\varepsilon\left(1+\frac{2p}{n-p}\right)$	$\varepsilon\left(1+\frac{12.2}{\delta^k_{\min}}\right)$	$\varepsilon\left(1+\frac{4}{\sigma^3_{\min}}\right)$
$\mathcal{A}_{\mathrm{SGD-samp}}$	–	–	$1.05\alpha\varepsilon^2\left(2\alpha+\frac{12.2^2}{(\delta^k_{\min})^2}\right)$	$2.1\alpha\varepsilon^2\left(\alpha+\frac{8}{\sigma^6_{\min}}\right)$
$\mathcal{A}_{\mathrm{SGD-iter}}$	$1.1\alpha\varepsilon^2\left(1+\frac{\eta^2p^2\frac{n}{\log n}}{\frac{n-p}{\log n-p}}\right)$	$1.1\alpha\varepsilon^2\left(1+\frac{4p^2\frac{n}{\log n}}{\frac{(n-p)^3}{\log n-p}}\right)$	–	–

DP Online Learning

Price of Privacy

Price of Privacy

Bounds for #mistakes

Price of Privacy

Bounds for #mistakes

	Offline Learning	Online Learning
Non-private algorithm		
Private algorithm		

Price of Privacy

Bounds for #mistakes

	Offline Learning	Online Learning
Non-private algorithm	$\Theta(\text{VCdim})$	
Private algorithm		

Price of Privacy

Bounds for #mistakes

	Offline Learning	Online Learning
Non-private algorithm	$\Theta(\mathrm{VCdim})$	$\Theta(\mathrm{Ldim})$
Private algorithm		

Price of Privacy

Bounds for #mistakes

	Offline Learning	Online Learning
Non-private algorithm	$\Theta(\text{VCdim})$	$\Theta(\text{Ldim})$
Private algorithm	$\leq \text{Ldim}^6$ [Ghazi et al., 2021]	

Price of Privacy

Bounds for #mistakes

	Offline Learning	Online Learning
Non-private algorithm	$\Theta(\text{VCdim})$	$\Theta(\text{Ldim})$
Private algorithm	$\leq \text{Ldim}^6$ [Ghazi et al., 2021]	$\leq O\left(2^{2^{\text{Ldim}}} \frac{1}{\varepsilon} \log T/\delta\right)$ [Golowich & Livni, 2021]

[3] Golowich, N., & Livni, R. (2021). 1&. *Advances in Neural Information Processing Systems*

Price of Privacy

Bounds for #mistakes

	Offline Learning	Online Learning
Non-private algorithm	$\Theta(\mathrm{VCdim})$	$\Theta(\mathrm{Ldim})$
Private algorithm	$\geq \log^*(\mathrm{Ldim})$ [Alon et al., 2022] $\leq \mathrm{Ldim}^6$ [Ghazi et al., 2021]	$\leq O\left(2^{2^{\mathrm{Ldim}}}\frac{1}{\varepsilon}\log T/\delta\right)$ [Golowich & Livni, 2021]

[1] Alon, N., Bun, M., Livni, R., Malliaris, M., & Moran, S. (2022). Private and online learnability are equivalent. *ACM Journal of the ACM (JACM)*
[2] Ghazi, B., Golowich, N., Kumar, R., & Manurangsi, P. (2021). Sample-efficient proper PAC learning with approximate differential privacy. In *Proceedings of the 53rd Annual ACM SIGACT Symposium on Theory of Computing*
[3] Golowich, N., & Livni, R. (2021). 1&. *Advances in Neural Information Processing Systems*

Price of Privacy

Bounds for #mistakes

	Offline Learning	Online Learning
Non-private algorithm	$\Theta(\mathrm{VCdim})$	$\Theta(\mathrm{Ldim})$
Private algorithm	$\geq \log^*(\mathrm{Ldim})$ [Alon et al., 2022] $\leq \mathrm{Ldim}^6$ [Ghazi et al., 2021]	$\geq ?$ $\leq O\left(2^{2^{\mathrm{Ldim}}}\frac{1}{\varepsilon}\log T/\delta\right)$ [Golowich & Livni, 2021]

[1] Alon, N., Bun, M., Livni, R., Malliaris, M., & Moran, S. (2022). Private and online learnability are equivalent. *ACM Journal of the ACM (JACM)*
[2] Ghazi, B., Golowich, N., Kumar, R., & Manurangsi, P. (2021). Sample-efficient proper PAC learning with approximate differential privacy.
In *Proceedings of the 53rd Annual ACM SIGACT Symposium on Theory of Computing*
[3] Golowich, N., & Livni, R. (2021). 1&. *Advances in Neural Information Processing Systems*

Price of Privacy

Price of Privacy

Previous result showed an upper bound, having a $\log T$ factor

Price of Privacy

Previous result showed an upper bound, having a $\log T$ factor

$$\leq O\left(2^{2^{\mathrm{Ldim}}}\frac{1}{\varepsilon}\log T/\delta\right)$$ [Golowich & Livni, 2021]

[3] Golowich, N., & Livni, R. (2021). Littlestone classes are privately online learnable. *Advances in Neural Information Processing Systems*

Price of Privacy

Previous result showed an upper bound, having a $\log T$ factor

$$\leq O\left(2^{2^{\mathrm{Ldim}}}\frac{1}{\varepsilon}\log T/\delta\right)$$ [Golowich & Livni, 2021]

[3] Golowich, N., & Livni, R. (2021). Littlestone classes are privately online learnable. *Advances in Neural Information Processing Systems*

Price of Privacy

Previous result showed an upper bound, having a $\log T$ factor

$$\leq O\left(2^{2^{\mathrm{Ldim}}} \frac{1}{\varepsilon} \log T / \delta\right)$$ [Golowich & Livni, 2021]

Recall that, for **non-private** online learning, there is no dependency on T

[3] Golowich, N., & Livni, R. (2021). Littlestone classes are privately online learnable. *Advances in Neural Information Processing Systems*

Price of Privacy

Previous result showed an upper bound, having a $\log T$ factor

$$\leq O\left(2^{2^{\mathrm{Ldim}}} \frac{1}{\varepsilon} \log T/\delta\right)$$ [Golowich & Livni, 2021]

Recall that, for **non-private** online learning, there is no dependency on T

Q: Does the expected #mistakes inherently increase with T for private online learning?

[3] Golowich, N., & Livni, R. (2021). Littlestone classes are privately online learnable. *Advances in Neural Information Processing Systems*

Simple online game

Simple online game

$$\mathcal{X} = \{0, 1\}$$

$$F = \left\{ f^{(0)}, f^{(1)} \right\}$$

Simple online game

$$\mathcal{X} = \{0, 1\}$$

$$F = \left\{ f^{(0)}, f^{(1)} \right\}$$

	Value at 0	Value at 1
$f^{(0)}$	0	0
$f^{(1)}$	0	1

Simple online game

- Learner $\mathcal{A}$ guesses between $f^{(0)}$ and $f^{(1)}$

$$\mathcal{X} = \{0, 1\}$$

$$F = \left\{ f^{(0)}, f^{(1)} \right\}$$

	Value at 0	Value at 1
$f^{(0)}$	0	0
$f^{(1)}$	0	1

Simple online game

$$\mathcal{X} = \{0, 1\}$$

$$F = \left\{ f^{(0)}, f^{(1)} \right\}$$

- Learner $\mathcal{A}$ guesses between $f^{(0)}$ and $f^{(1)}$
- Adversary chooses $x \in \{0, 1\}$ s.t. $f^* = f^{(x)}$

	Value at 0	Value at 1
$f^{(0)}$	0	0
$f^{(1)}$	0	1

Simple online game

$$\mathcal{X} = \{0, 1\}$$

$$F = \left\{ f^{(0)}, f^{(1)} \right\}$$

- Learner $\mathcal{A}$ guesses between $f^{(0)}$ and $f^{(1)}$
- Adversary chooses $x \in \{0, 1\}$ s.t. $f^* = f^{(x)}$
- Adversary can only give $(0, 0)$ and $(1, x)$

	Value at 0	Value at 1
$f^{(0)}$	0	0
$f^{(1)}$	0	1

Simple online game

$$\mathcal{X} = \{0, 1\}$$

$$F = \left\{f^{(0)}, f^{(1)}\right\}$$

- Learner $\mathcal{A}$ guesses between $f^{(0)}$ and $f^{(1)}$
- Adversary chooses $x \in \{0, 1\}$ s.t. $f^* = f^{(x)}$
- Adversary can only give $(0, 0)$ and $(1, x)$
- Denote $\perp := (0, 0)$ and $x := (1, x)$

	Value at 0	Value at 1
$f^{(0)}$	0	0
$f^{(1)}$	0	1

Simple online game

$$\mathcal{X} = \{0, 1\}$$

$$F = \left\{ f^{(0)}, f^{(1)} \right\}$$

- Learner $\mathcal{A}$ guesses between $f^{(0)}$ and $f^{(1)}$
- Adversary chooses $x \in \{0, 1\}$ s.t. $f^* = f^{(x)}$
- Adversary can only give $(0, 0)$ and $(1, x)$
- Denote $\perp := (0, 0)$ and $x := (1, x)$

	Value at 0	Value at 1
$f^{(0)}$	0	0
$f^{(1)}$	0	1

$\exists \mathcal{A}$ s.t. $M = 1$

Starter: Throw Away algorithm (Name & Shame)

Starter: Throw Away algorithm (Name & Shame)

- $S \leftarrow \emptyset$

Starter: Throw Away algorithm (Name & Shame)

- $S \leftarrow \emptyset$
- For $t \in [T]$:

Starter: Throw Away algorithm (Name & Shame)

- $S \leftarrow \emptyset$
- For $t \in [T]$:
  - If S contains $x \neq \perp$, **output** x

Starter: Throw Away algorithm (Name & Shame)

- $S \leftarrow \varnothing$
- For $t \in [T]$:
    - If S contains $x \neq \perp$, **output** x
    - Else, **output 0**

Starter: Throw Away algorithm (Name & Shame)

- $S \leftarrow \varnothing$
- For $t \in [T]$:
    - If S contains $x \neq \perp$, **output** x
    - Else, **output 0**
    - Add new sample to S with probability δ

Starter: Throw Away algorithm (Name & Shame)

- $S \leftarrow \varnothing$
- For $t \in [T]$:
  - If S contains $x \neq \perp$, **output** x
  - Else, **output 0**
  - Add new sample to S with probability δ

Algorithm is 'private' and will make $\frac{1}{\delta}$ mistakes in expectation. Can be generalized to arbitrary classes, with $\frac{\mathrm{Ldim}(F)}{\delta}$ mistakes.

Starter: Throw Away algorithm (Name & Shame)

- $S \leftarrow \emptyset$
- For $t \in [T]$:
    - If S contains $x \neq \perp$, **output** x
    - Else, **output 0**
    - Add new sample to S with probability δ

Algorithm is 'private' and will make $\frac{1}{\delta}$ mistakes in expectation. Can be generalized to arbitrary classes, with $\frac{\mathrm{Ldim}(F)}{\delta}$ mistakes.

But this completely leaks the privacy of all points in S.

β-Concentrated Algorithm

β-Concentrated Algorithm

An algorithm $\mathbb{A}$ is β-*concentrated*, if $\exists x \in \{0,1\}$, such that if $\mathbb{P}(\mathbb{A}((\perp, \perp, \ldots, \perp)) = (x, x, \ldots, x)) \geq 1 - \beta$

β-Concentrated Algorithm

An algorithm $\mathbb{A}$ is β-*concentrated*, if $\exists x \in \{0,1\}$, such that if $\mathbb{P}(\mathbb{A}((\perp, \perp, \ldots, \perp)) = (x, x, \ldots, x)) \geq 1 - \beta$

$y_t : \perp$

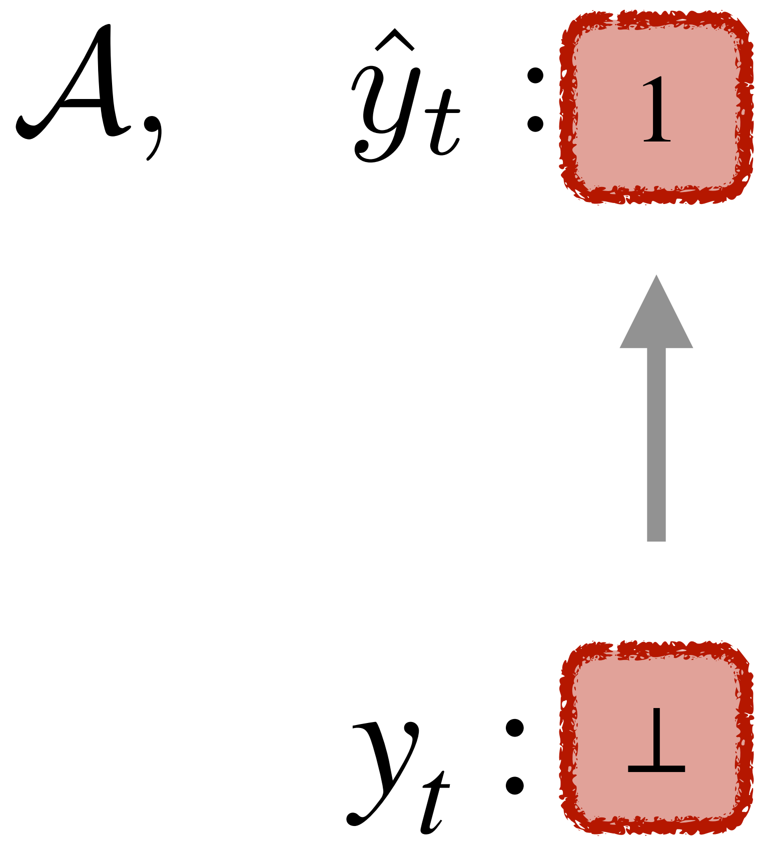
β-Concentrated Algorithm

An algorithm $\mathbb{A}$ is β-*concentrated*, if $\exists x \in \{0,1\}$, such that if
$\mathbb{P}(\mathbb{A}((\perp, \perp, \ldots, \perp)) = (x, x, \ldots, x)) \geq 1 - \beta$

$y_t : \perp$

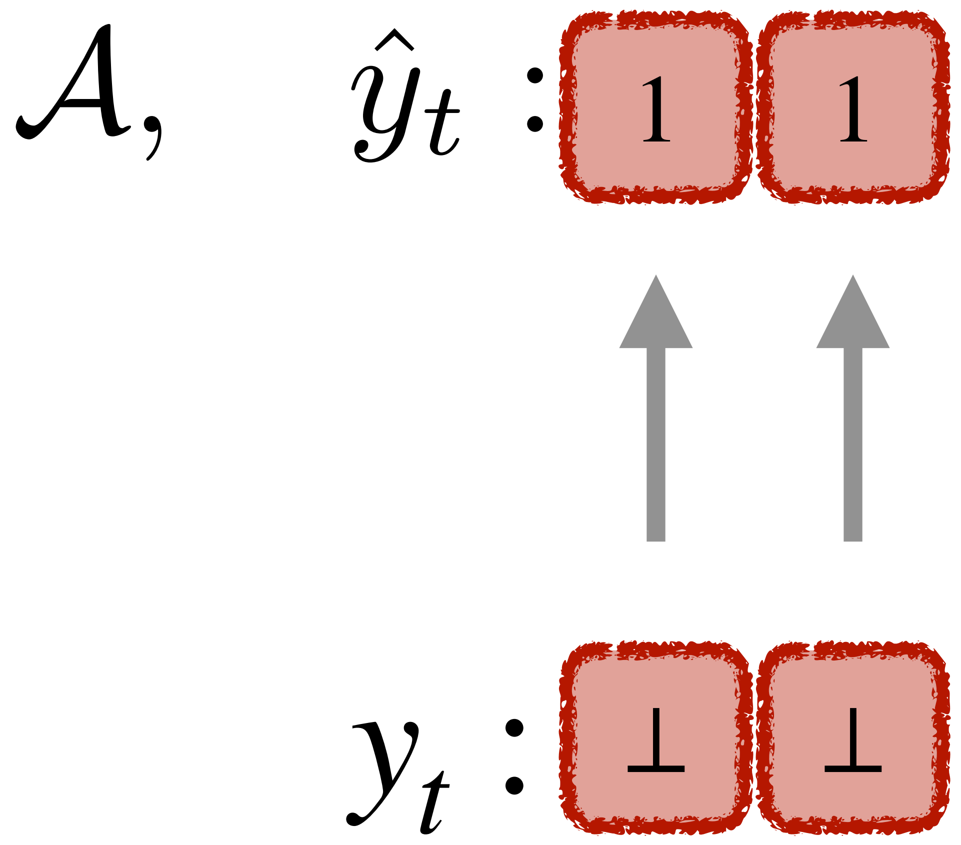
β-Concentrated Algorithm

An algorithm $\mathbb{A}$ is *β-concentrated*, if $\exists x \in \{0,1\}$, such that if $\mathbb{P}(\mathbb{A}((\perp, \perp, \ldots, \perp)) = (x, x, \ldots, x)) \geq 1 - \beta$



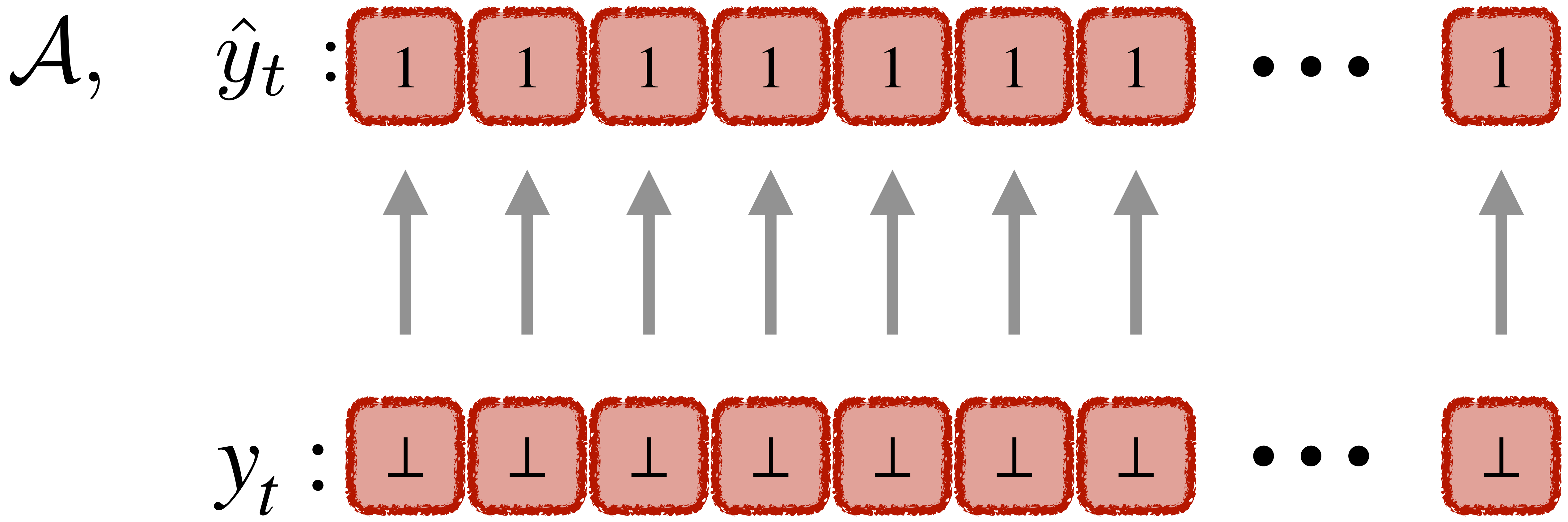
β-Concentrated Algorithm

An algorithm $\mathbb{A}$ is *β-concentrated*, if $\exists x \in \{0,1\}$, such that if
$\mathbb{P}(\mathbb{A}((\perp, \perp, \ldots, \perp)) = (x, x, \ldots, x)) \geq 1 - \beta$



β-Concentrated Algorithm

An algorithm $\mathbb{A}$ is *β-concentrated*, if $\exists x \in \{0,1\}$, such that if
$\mathbb{P}(\mathbb{A}((\perp, \perp, \ldots, \perp)) = (x, x, \ldots, x)) \geq 1 - \beta$



β-Concentrated Algorithm

$$s := \mathcal{A}(\perp, \ldots, \perp)$$

$$\mathbb{P}(s = (1, 1, \ldots, 1)) \geq 1 - \beta$$

β-Concentrated Algorithm

$$s := \mathcal{A}(\perp, \ldots, \perp)$$

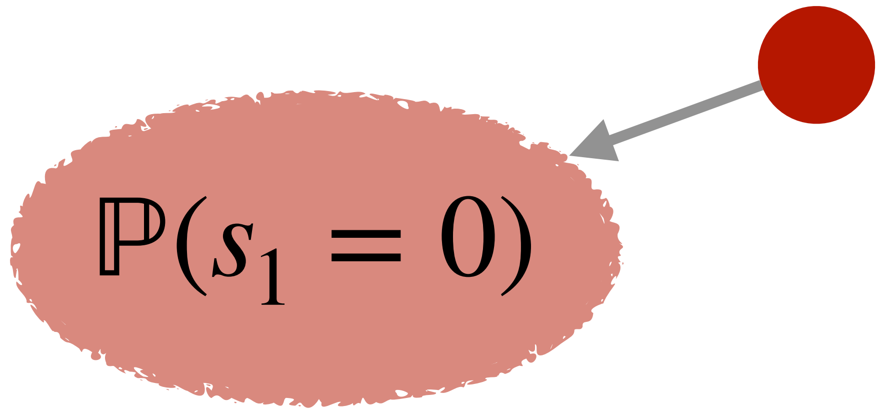
$$\mathbb{P}(s = (1, 1, \ldots, 1)) \geq 1 - \beta$$

β-Concentrated Algorithm

$$s := \mathcal{A}(\perp, \ldots, \perp)$$

$$\mathbb{P}(s = (1, 1, \ldots, 1)) \geq 1 - \beta$$

$t = 1$

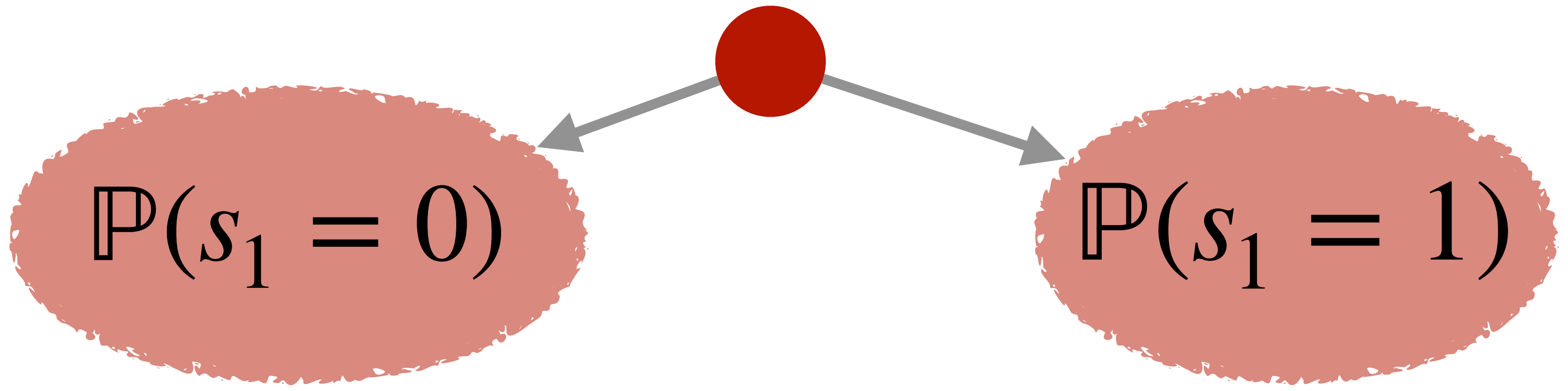


β-Concentrated Algorithm

$$s := \mathcal{A}(\perp, \ldots, \perp)$$

$$\mathbb{P}(s = (1, 1, \ldots, 1)) \geq 1 - \beta$$

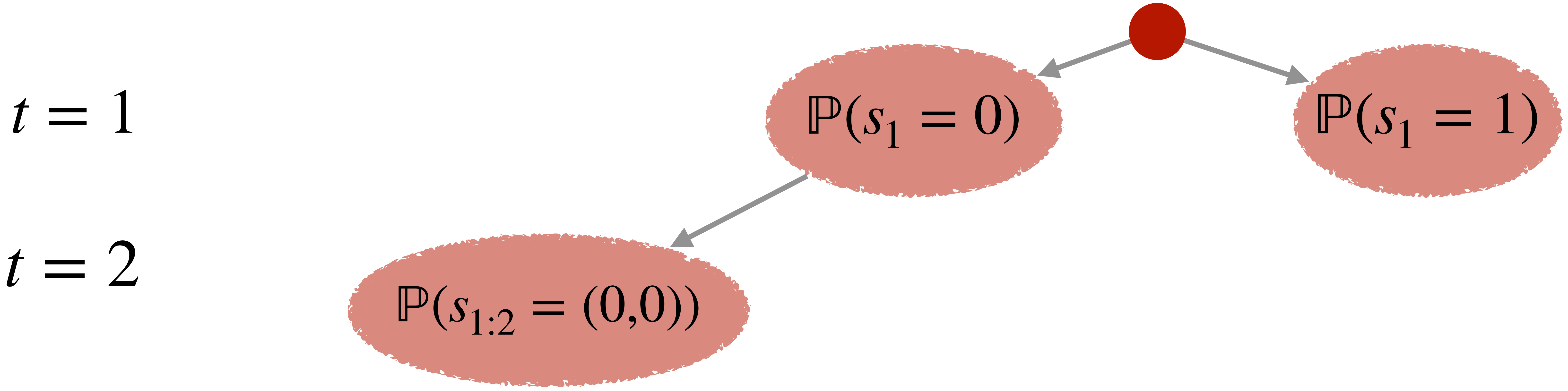
$t = 1$



β-Concentrated Algorithm

$$s := \mathcal{A}(\bot, \ldots, \bot)$$

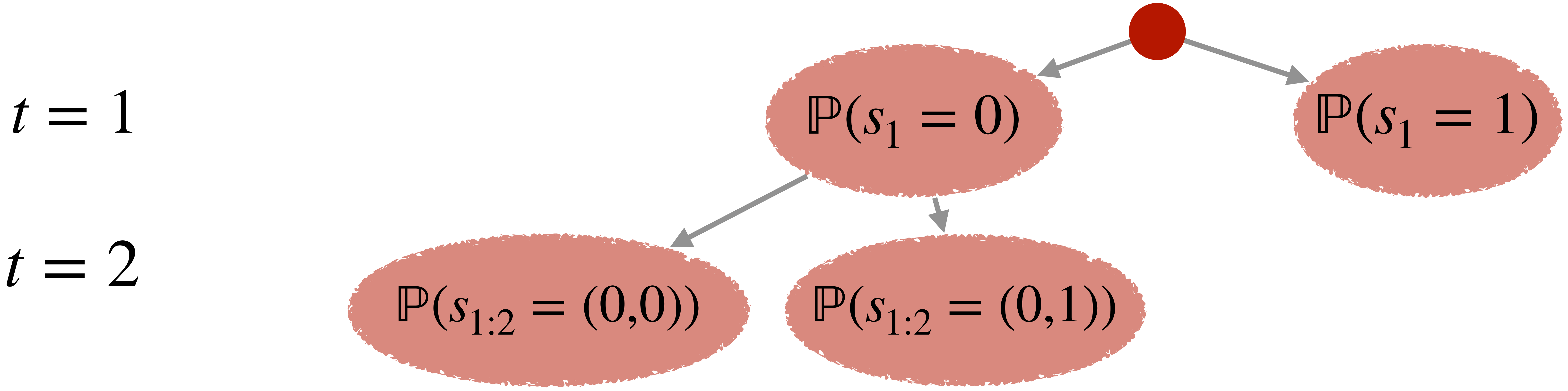
$$\mathbb{P}(s = (1, 1, \ldots, 1)) \geq 1 - \beta$$



β-Concentrated Algorithm

$$s := \mathcal{A}(\perp, \ldots, \perp)$$

$$\mathbb{P}(s = (1, 1, \ldots, 1)) \geq 1 - \beta$$



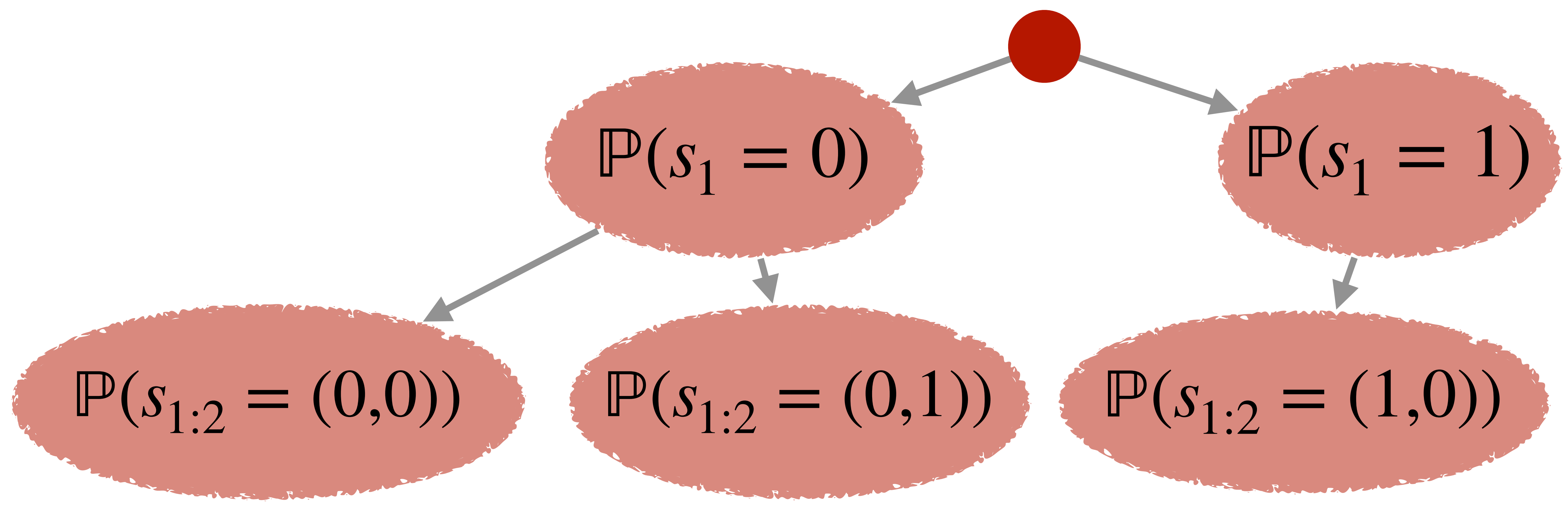
β-Concentrated Algorithm

$$s := \mathcal{A}(\perp, \ldots, \perp)$$

$$\mathbb{P}(s = (1, 1, \ldots, 1)) \geq 1 - \beta$$

$t = 1$

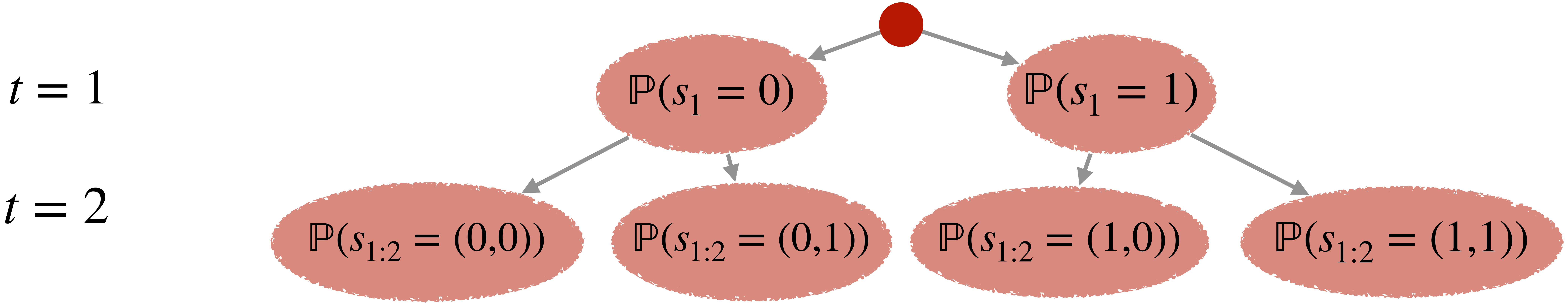
$t = 2$



β-Concentrated Algorithm

$$s := \mathcal{A}(\perp, \ldots, \perp)$$

$$\mathbb{P}(s = (1, 1, \ldots, 1)) \geq 1 - \beta$$



β-Concentrated Algorithm

$$s := \mathcal{A}(\perp, \ldots, \perp)$$

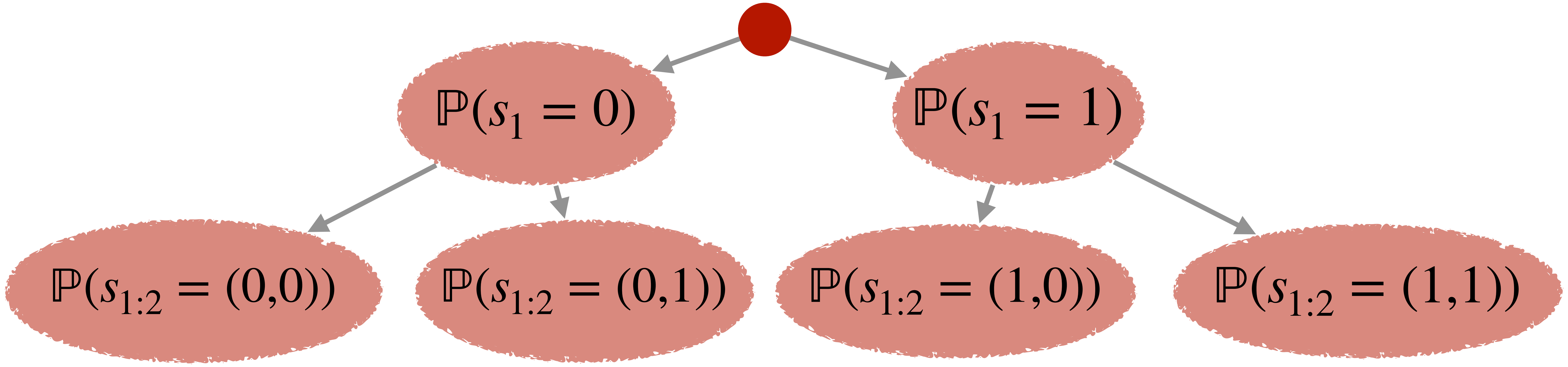
$$\mathbb{P}(s = (1, 1, \ldots, 1)) \geq 1 - \beta$$

$t = 1$

$t = 2$

$\vdots$

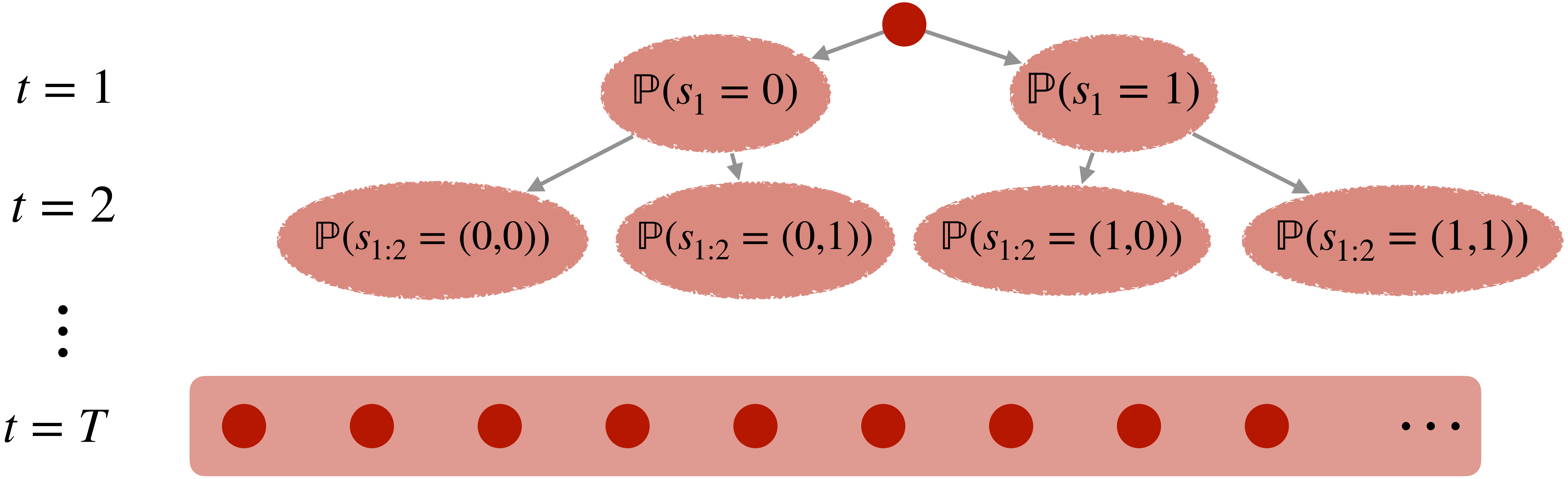
$t = T$



β-Concentrated Algorithm

$$s := \mathcal{A}(\perp, \ldots, \perp)$$

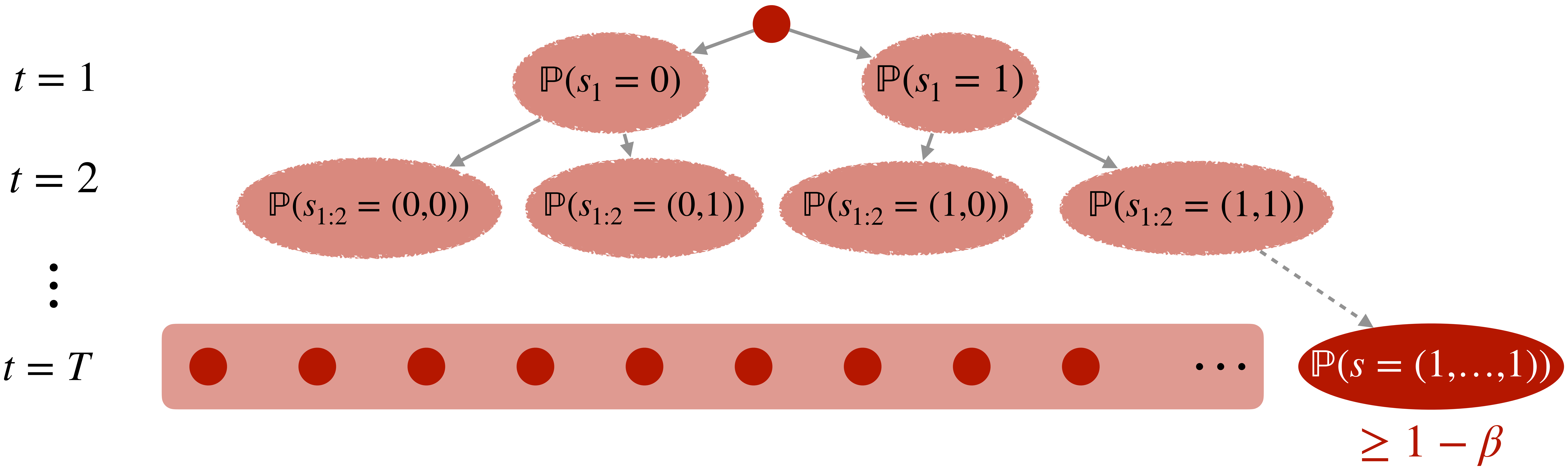
$$\mathbb{P}(s = (1, 1, \ldots, 1)) \geq 1 - \beta$$



β-Concentrated Algorithm

$$s := \mathcal{A}(\perp, \ldots, \perp)$$

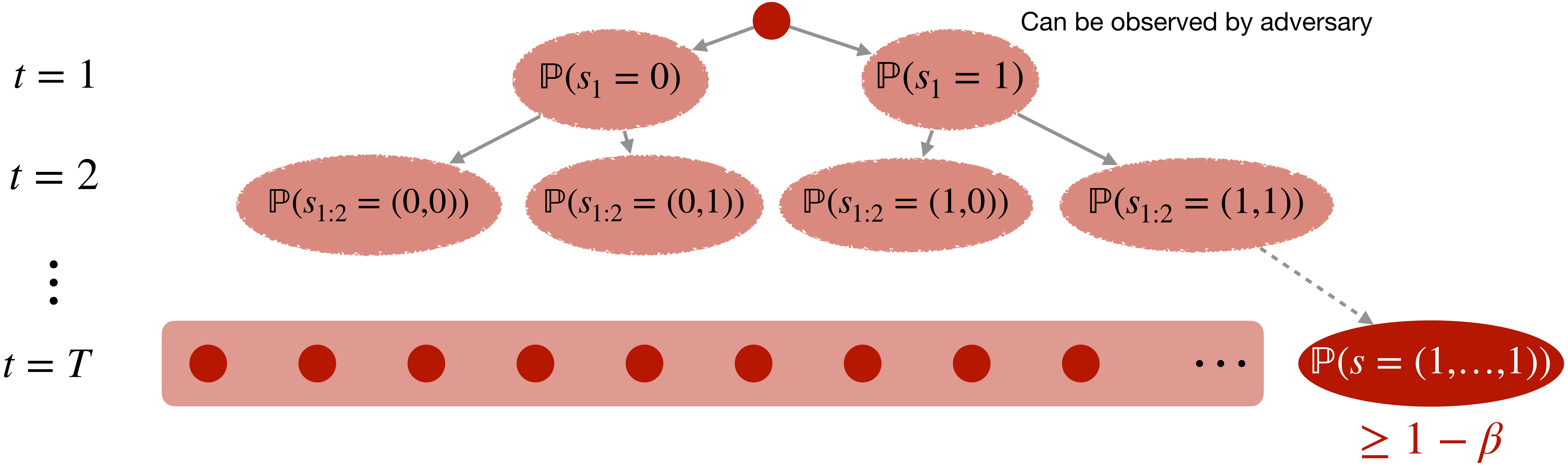
$$\mathbb{P}(s = (1, 1, \ldots, 1)) \geq 1 - \beta$$



β-Concentrated Algorithm

$$s := \mathcal{A}(\perp, \ldots, \perp)$$

$$\mathbb{P}(s = (1, 1, \ldots, 1)) \geq 1 - \beta$$



Main Result

Main Result

An algorithm $\mathbb{A}$ is β-*concentrated*, if $\exists x \in \{0,1\}$, such that if
$\mathbb{P}(\mathbb{A}((\perp, \perp, \ldots, \perp)) = (x, x, \ldots, x)) \geq 1 - \beta$

Main Result

An algorithm $\mathbb{A}$ is β-*concentrated*, if $\exists x \in \{0,1\}$, such that if
$$\mathbb{P}(\mathbb{A}((\perp, \perp, \ldots, \perp)) = (x, x, \ldots, x)) \geq 1 - \beta$$

Theorem

Main Result

An algorithm $\mathbb{A}$ is β-*concentrated*, if $\exists x \in \{0,1\}$, such that if $\mathbb{P}(\mathbb{A}((\perp, \perp, \ldots, \perp)) = (x, x, \ldots, x)) \geq 1 - \beta$

Theorem

For any $\varepsilon, \delta > 0$, for any 0.1-*concentrated* $\mathcal{A}$, there exists an adversary, such that

Main Result

An algorithm $\mathbb{A}$ is *β-concentrated*, if $\exists x \in \{0,1\}$, such that if
$\mathbb{P}(\mathbb{A}((\perp, \perp, \ldots, \perp)) = (x, x, \ldots, x)) \geq 1 - \beta$

Theorem

For any $\varepsilon, \delta > 0$, for any 0.1-*concentrated* $\mathcal{A}$, there exists an adversary, such that

$$\text{For,} \quad T \leq \exp(1/16\delta), \ \mathbb{E}\left[\mathrm{M}\right] = \widetilde{\Omega}\left(\frac{\log T/\delta}{\varepsilon}\right)$$

Main Result

An algorithm $\mathbb{A}$ is *β-concentrated*, if $\exists x \in \{0,1\}$, such that if

$$\mathbb{P}(\mathbb{A}((\perp, \perp, \ldots, \perp)) = (x, x, \ldots, x)) \geq 1 - \beta$$

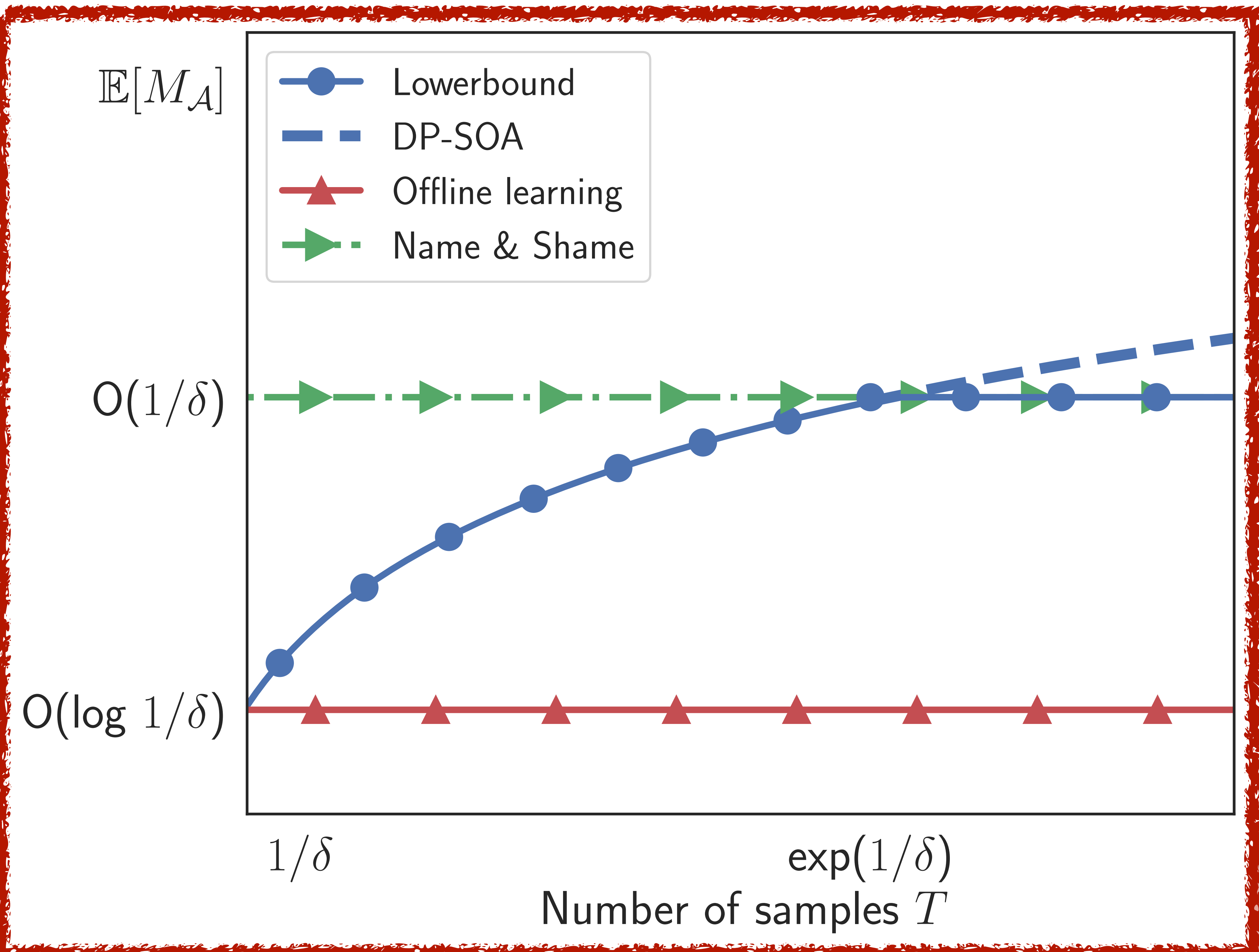
Theorem

For any $\varepsilon, \delta > 0$, for any 0.1-*concentrated* $\mathcal{A}$, there exists an adversary, such that

$$\text{For,} \quad T \leq \exp(1/16\delta),\ \mathbb{E}[\mathrm{M}] = \widetilde{\Omega}\left(\frac{\log T/\delta}{\varepsilon}\right)$$

$$\text{For,} \quad T > \exp(1/16\delta),\ \mathbb{E}[\mathrm{M}] = \widetilde{\Omega}\left(\frac{1}{\delta}\right)$$

Main Result



Theorem

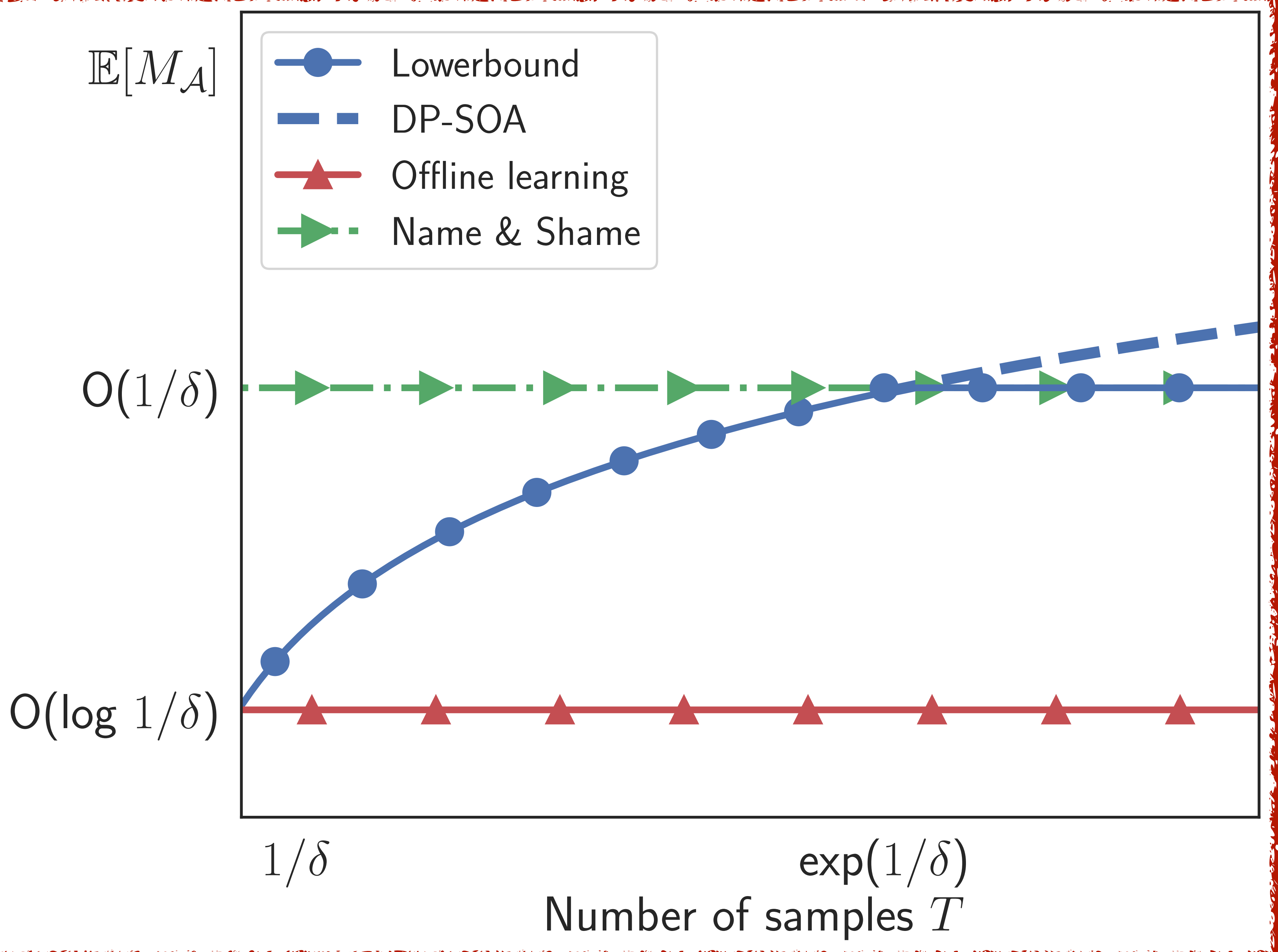
For any $\varepsilon, \delta > 0$, for any 0.1-*concentrated* $\mathcal{A}$, there exists an adversary, such that

$$\text{For,} \quad T \leq \exp(1/16\delta),\ \mathbb{E}\left[\mathrm{M}\right] = \widetilde{\Omega}\left(\frac{\log T/\delta}{\varepsilon}\right)$$

$$\text{For,} \quad T > \exp(1/16\delta),\ \mathbb{E}\left[\mathrm{M}\right] = \widetilde{\Omega}\left(\frac{1}{\delta}\right)$$

Main Result

Proof technique:
Truly online lower bound



Theorem

For any $\varepsilon, \delta > 0$, for any 0.1-*concentrated* $\mathcal{A}$, there exists an adversary, such that

$$\text{For,} \quad T \leq \exp(1/16\delta), \ \mathbb{E}\left[\mathrm{M}\right] = \widetilde{\Omega}\left(\frac{\log T/\delta}{\varepsilon}\right)$$

$$\text{For,} \quad T > \exp(1/16\delta), \ \mathbb{E}\left[\mathrm{M}\right] = \widetilde{\Omega}\left(\frac{1}{\delta}\right)$$

Corollary

There exists a function class, Point_N, such that any (ε, δ)-private proper online algorithm must have $\mathbb{E}\left[\mathrm{M}\right] \geq \min\left(\frac{1}{\delta}, \frac{1}{1000\varepsilon}\log\frac{T}{\delta}\right)$ in the worst case.

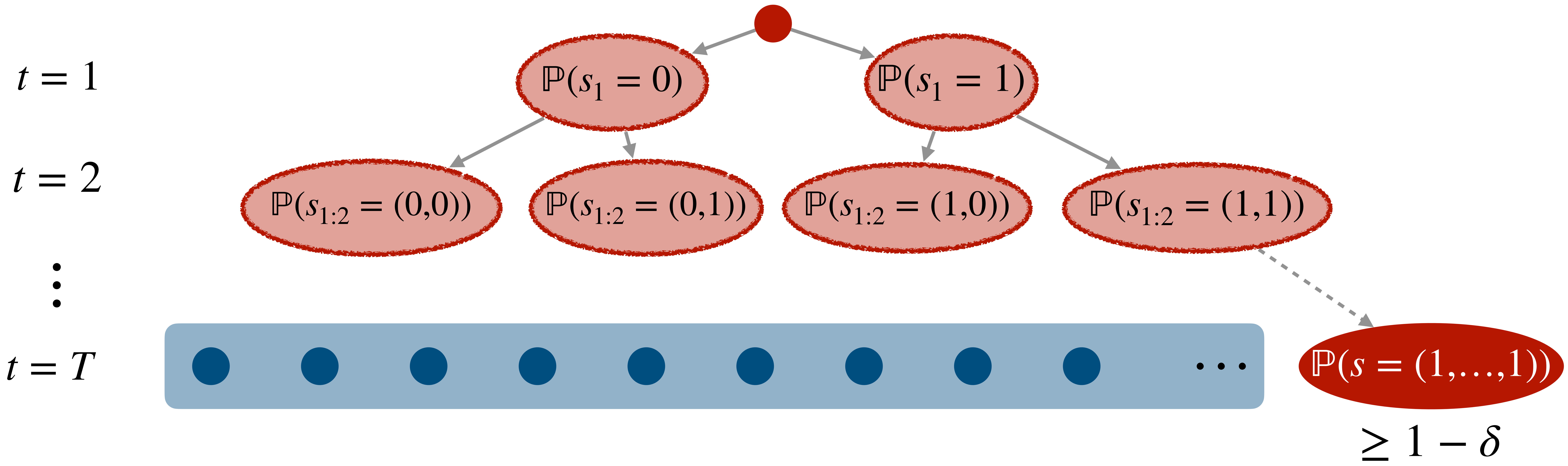
Concurrent work [CLNSS24]

For $N \geq \sqrt{T}$, Point_∞, any (ε, δ)-private online algorithm must have $\mathbb{E}\left[\mathrm{M}\right] \geq \min\left(\frac{1}{\delta}, \frac{1}{1000\varepsilon}\log T\right)$ in the worst case.

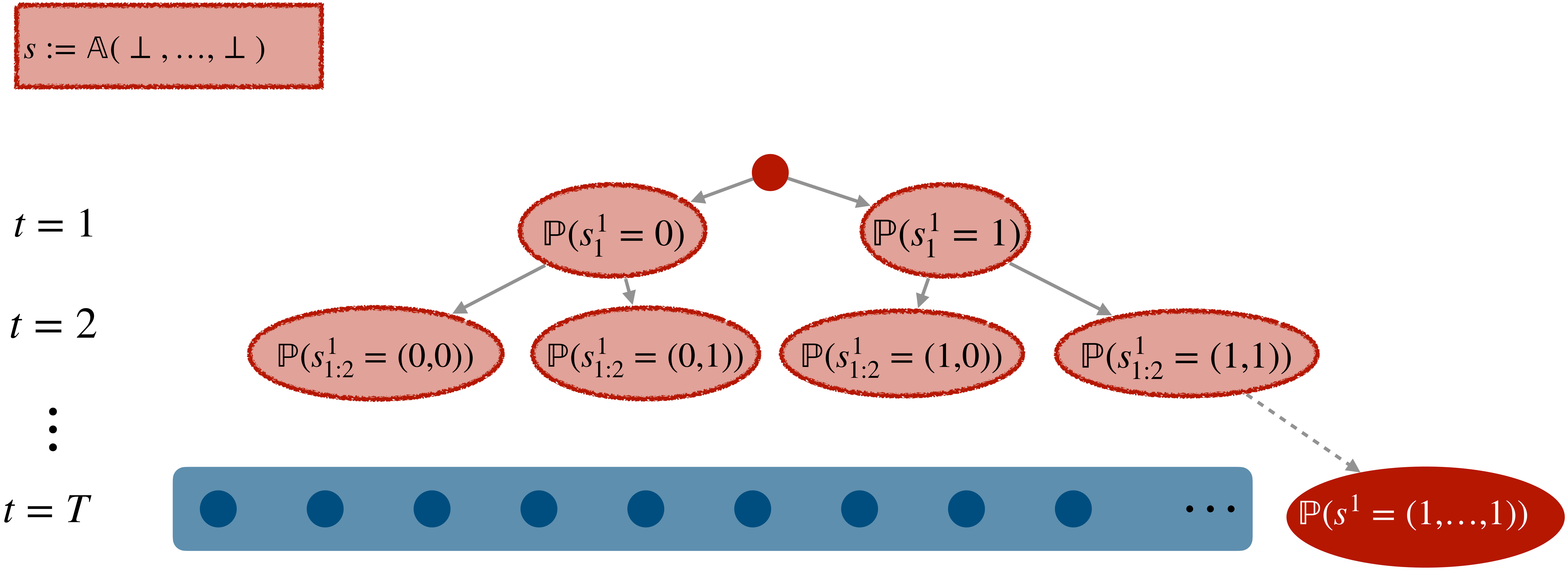
Proof Sketch

Concentration

$s := \mathbb{A}(\perp, \ldots, \perp)$



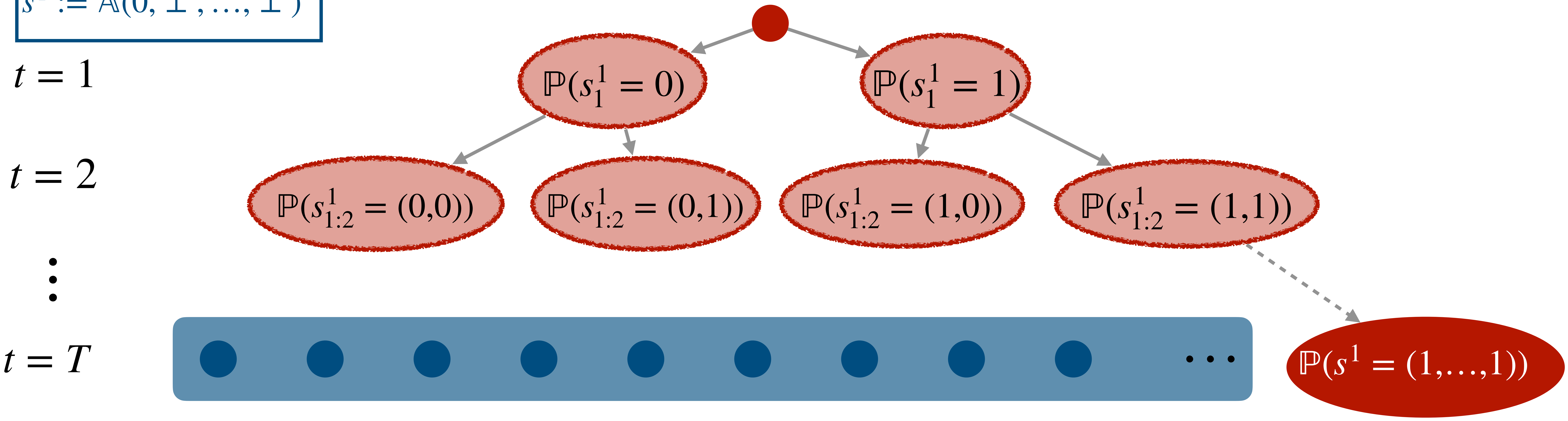
Applying DP on the `correct' event



Applying DP on the `correct' event

$s := \mathbb{A}(\perp, \ldots, \perp)$

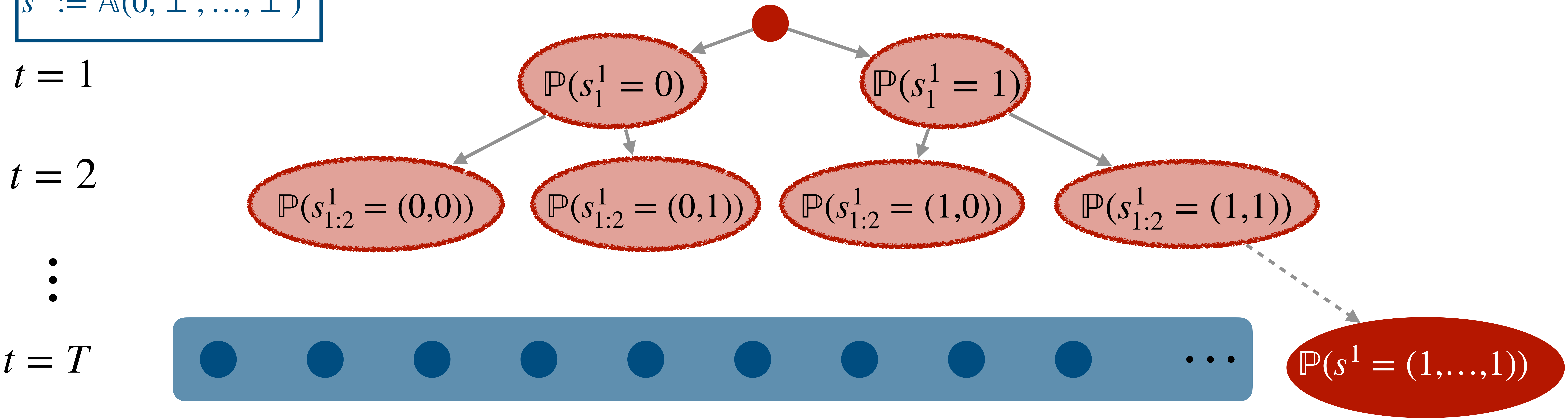
$s^1 := \mathbb{A}(0, \perp, \ldots, \perp)$



Applying DP on the `correct' event

$s := \mathbb{A}(\perp, \ldots, \perp)$

$s^1 := \mathbb{A}(0, \perp, \ldots, \perp)$

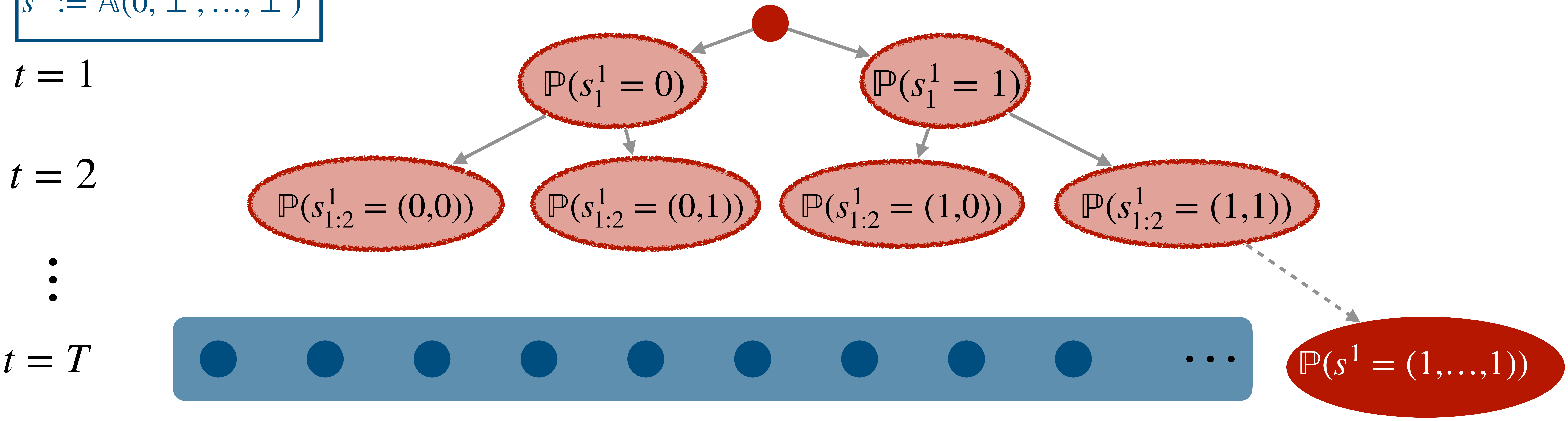


$$q_0 := \mathbb{P}\left(s \neq (1,\ldots,1)\right) \leq \delta$$

Applying DP on the `correct' event

$s := \mathbb{A}(\perp, \ldots, \perp)$

$s^1 := \mathbb{A}(0, \perp, \ldots, \perp)$



$q_0 := \mathbb{P}\left(s \neq (1,\ldots,1)\right) \leq \delta$

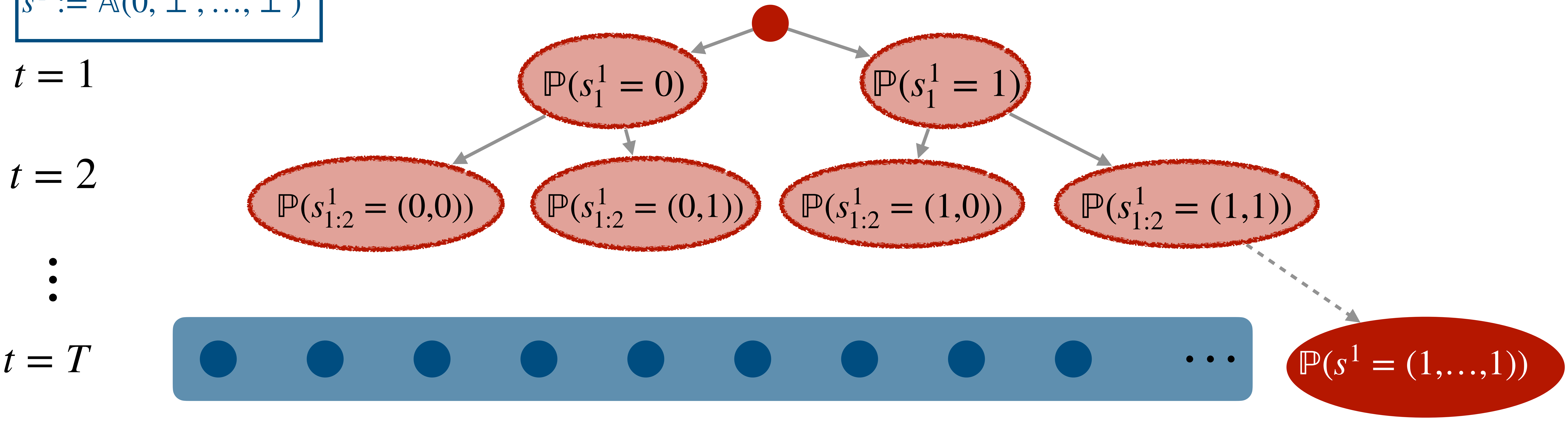
$\text{DP}: \mathbb{P}\left(s^1 \neq (1,\ldots,1)\right) \leq \exp(\varepsilon)\mathbb{P}\left(s \neq (1,\ldots,1)\right) + \delta$

Applying DP on the `correct' event

$s := \mathbb{A}(\perp, \ldots, \perp)$

$s^1 := \mathbb{A}(0, \perp, \ldots, \perp)$

Assume $\varepsilon = \ln \frac{3}{2}$



$q_0 := \mathbb{P}\left(s \neq (1,\ldots,1)\right) \leq \delta$

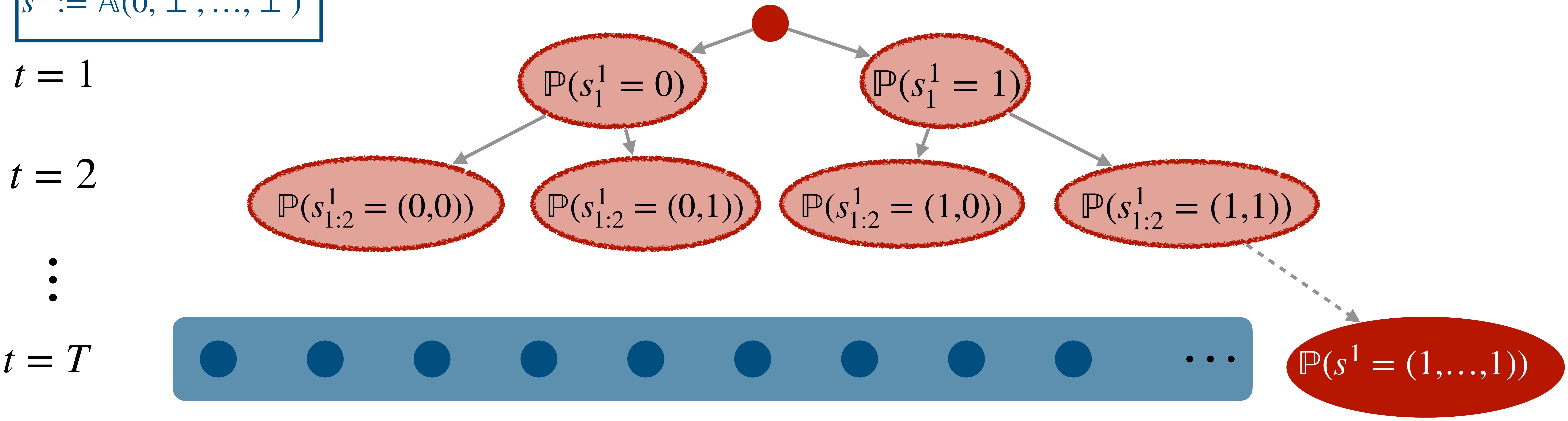
$\text{DP}: \mathbb{P}\left(s^1 \neq (1,\ldots,1)\right) \leq \exp(\varepsilon)\mathbb{P}\left(s \neq (1,\ldots,1)\right) + \delta$

Applying DP on the `correct' event

$s := \mathbb{A}(\perp, \ldots, \perp)$

$s^1 := \mathbb{A}(0, \perp, \ldots, \perp)$

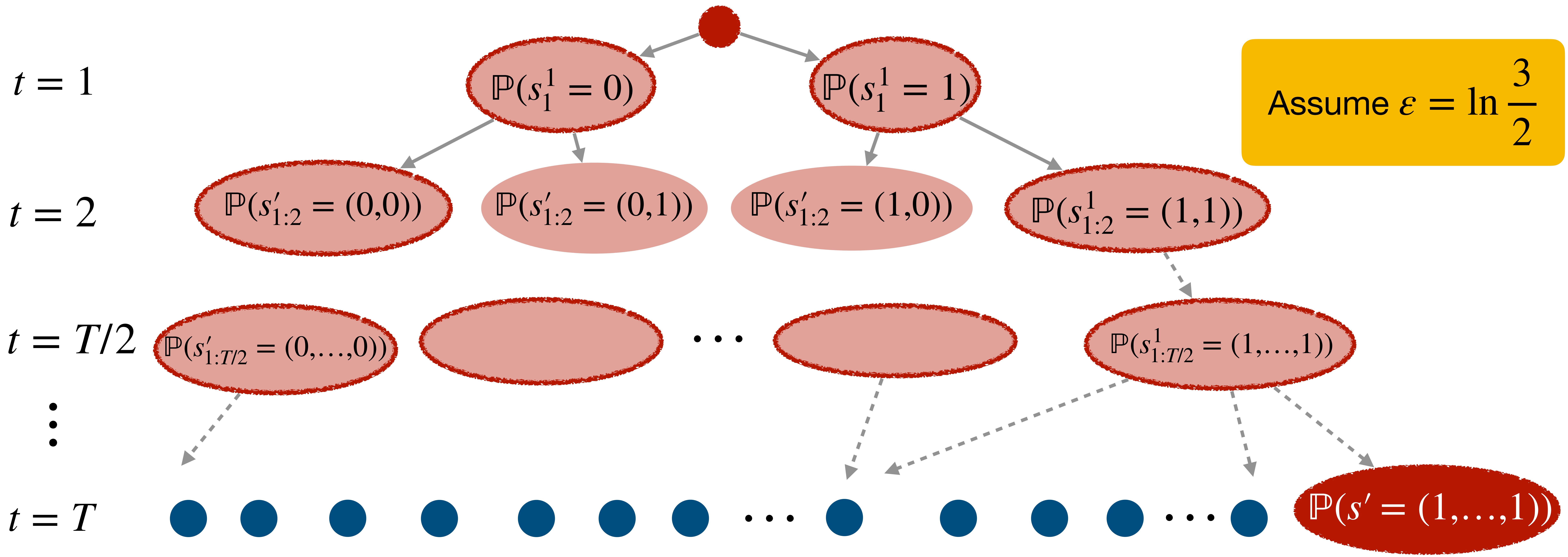
Assume $\varepsilon = \ln \frac{3}{2}$



$q_0 := \mathbb{P}\left(s \neq (1,\ldots,1)\right) \leq \delta$

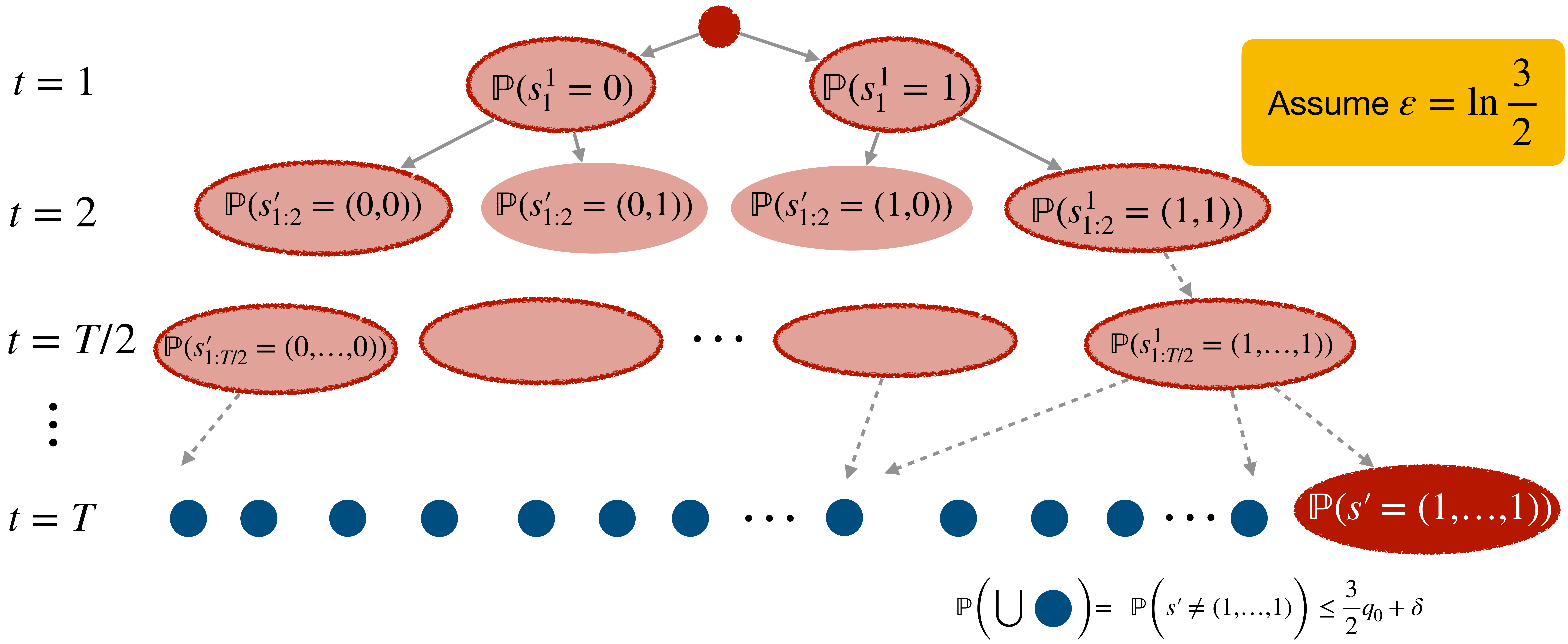
$\text{DP}: \mathbb{P}\left(s^1 \neq (1,\ldots,1)\right) \leq \exp(\varepsilon)\mathbb{P}\left(s \neq (1,\ldots,1)\right) + \delta = \frac{3}{2}q_0 + \delta$

Applying DP on the `correct' set (cont.)

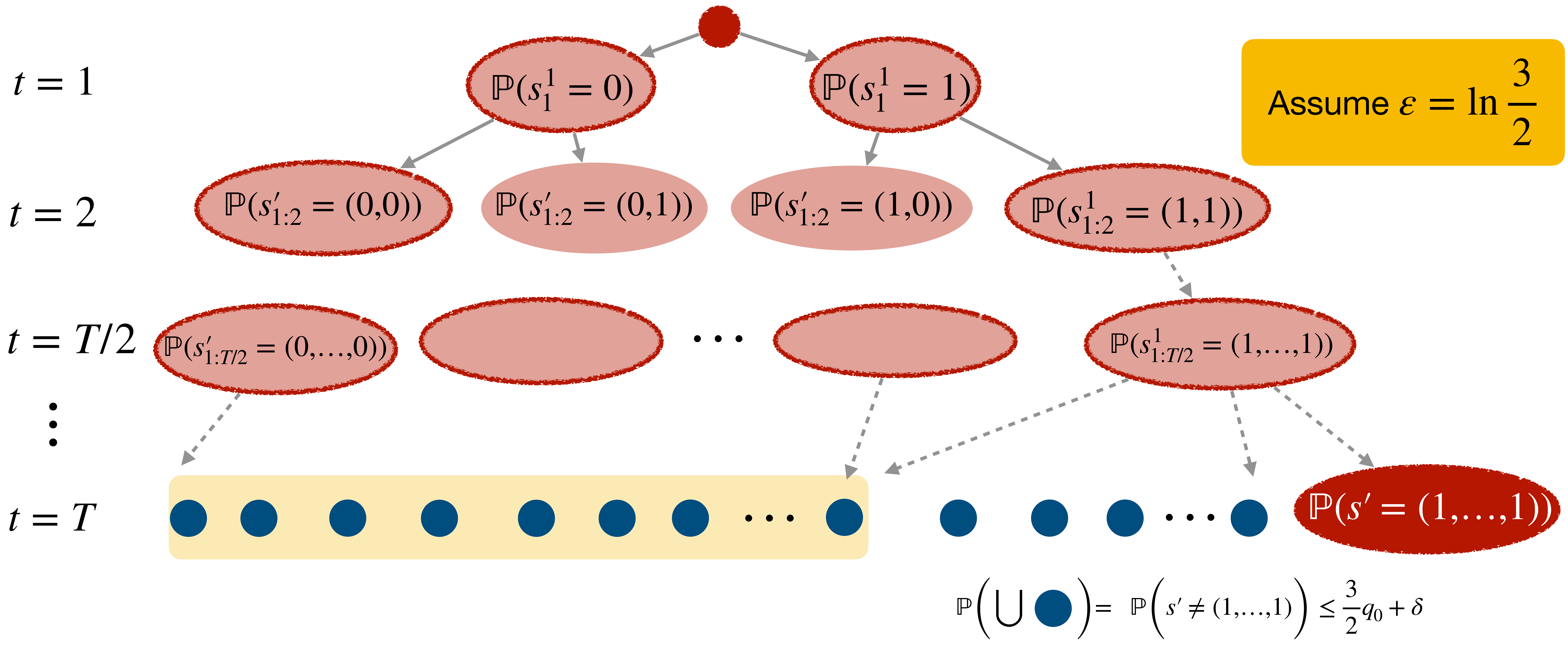


Assume $\varepsilon = \ln \frac{3}{2}$

Applying DP on the `correct' set (cont.)

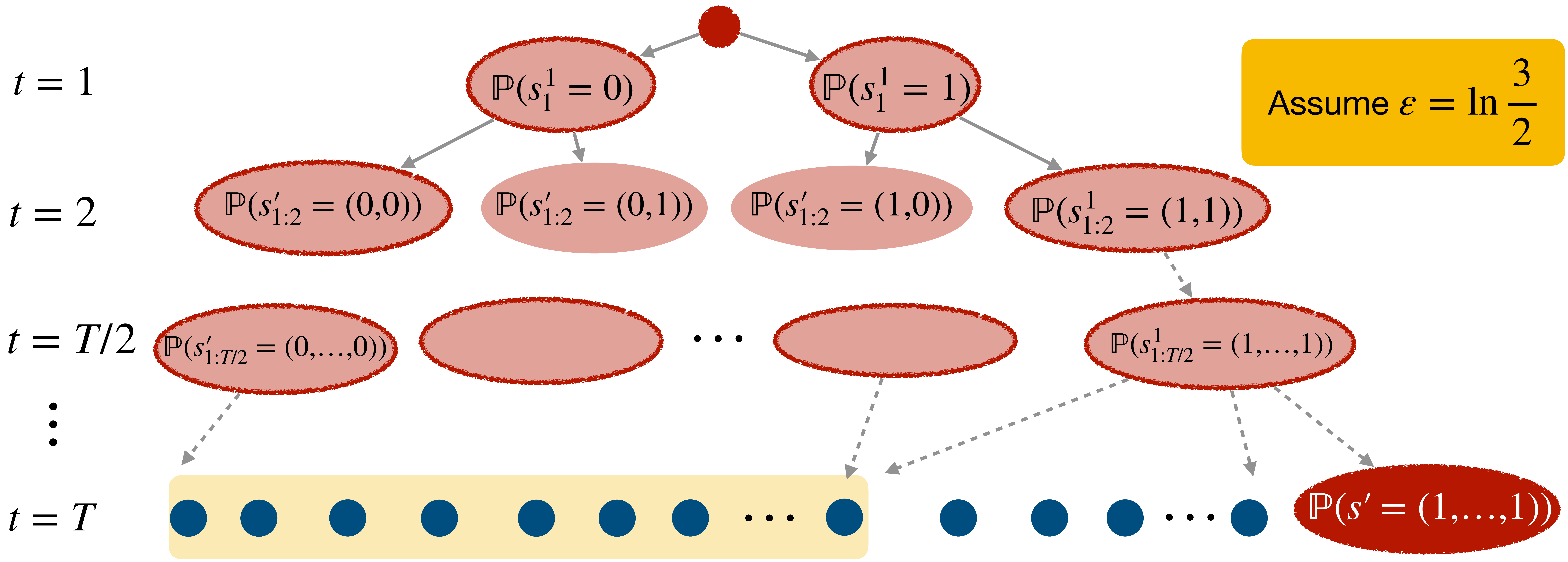


Applying DP on the `correct' set (cont.)



Applying DP on the `correct' set (cont.)

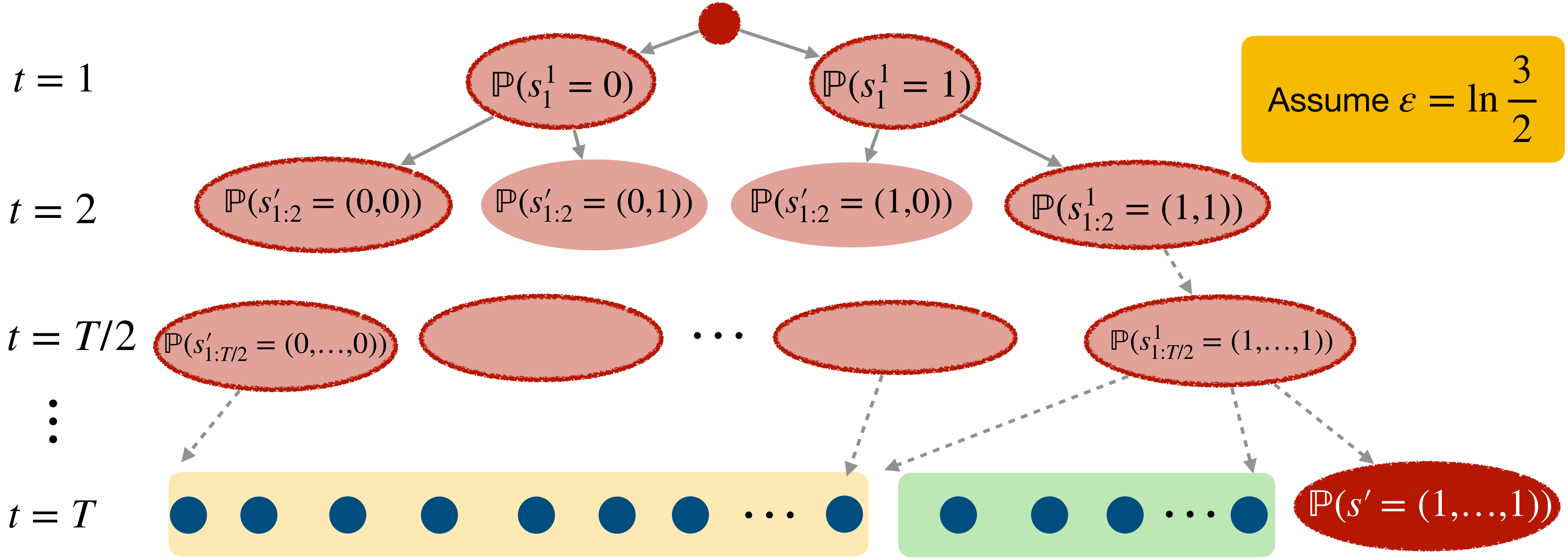
Assume $\varepsilon = \ln \frac{3}{2}$



I are the sequences containing '0' at least once before $t = T/2$

$$\mathbb{P}\left(\bigcup \bullet\right) = \mathbb{P}\left(s' \neq (1,\ldots,1)\right) \leq \frac{3}{2} q_0 + \delta$$

Applying DP on the `correct' set (cont.)

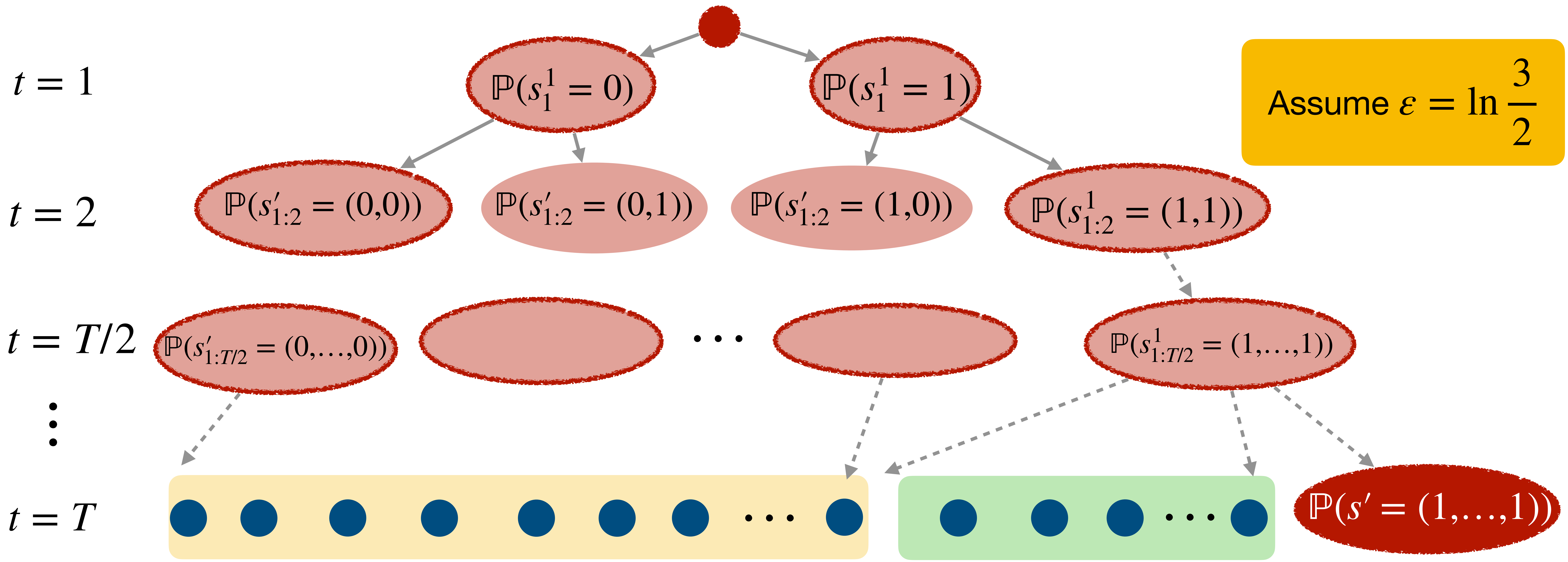


Assume $\varepsilon = \ln \frac{3}{2}$

I are the sequences containing '0' at least once before $t = T/2$

$\mathbb{P}\left(\bigcup \bullet\right) = \mathbb{P}\left(s' \neq (1,\ldots,1)\right) \leq \frac{3}{2} q_0 + \delta$

Applying DP on the `correct' set (cont.)



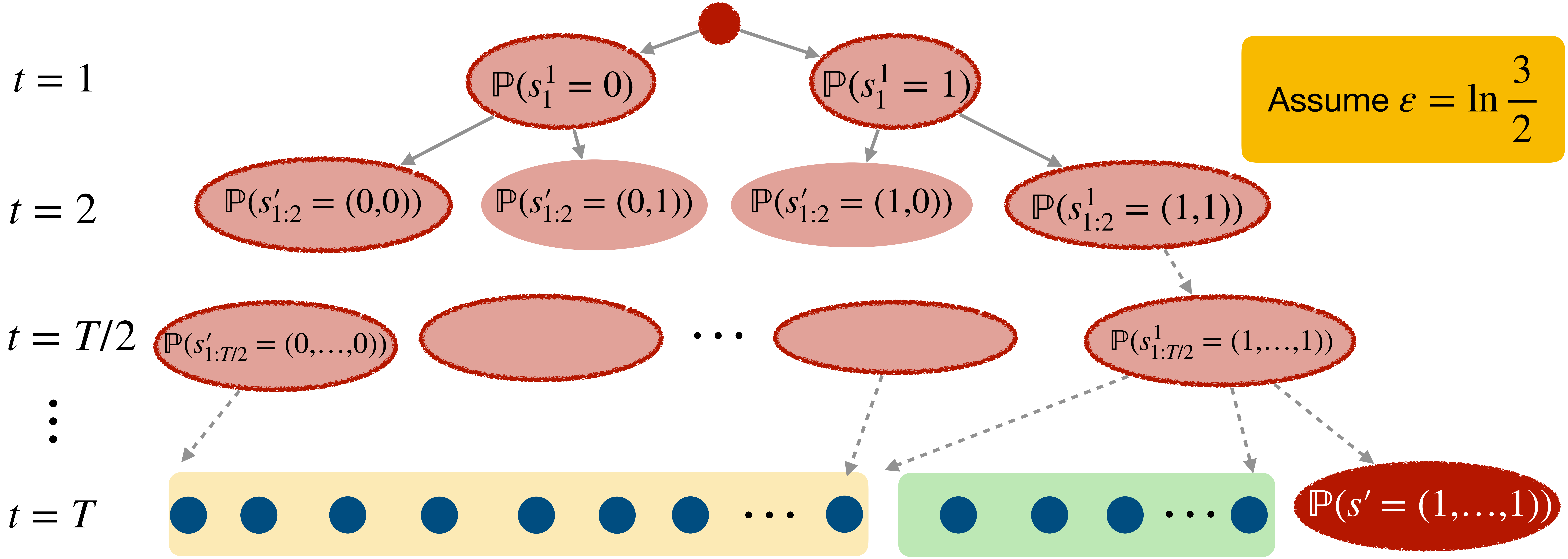
I are the sequences containing '0' at least once before $t = T/2$

$\mathbb{P}\left(\bigcup \bullet\right) = \mathbb{P}\left(s' \neq (1,\ldots,1)\right) \leq \frac{3}{2} q_0 + \delta$

II are the sequences containing only '1's until $t = T/2$ and containing '0' at least once when $T/2 < t \leq T$

Applying DP on the `correct' set (cont.)

Assume $\varepsilon = \ln \frac{3}{2}$



I are the sequences containing '0' at least once before $t = T/2$

II are the sequences containing only '1's until $t = T/2$ and containing '0' at least once when $T/2 < t \leq T$

$$\mathbb{P}\left(\bigcup \bullet\right) = \mathbb{P}\left(s' \neq (1,\ldots,1)\right) \leq \frac{3}{2}q_0 + \delta$$

$$\mathbb{P}\left(\bigcup \bullet\right) = \mathbb{P}\left(\text{I}\right) + \mathbb{P}\left(\text{II}\right) \leq \frac{3}{2}q_0 + \delta$$

Recursion

Recursion

I are the sequences containing ‘0’ at least once before $t = T/2$

Recursion

I are the sequences containing ‘0’ at least once before $t = T/2$

II are the sequences containing only ‘1’s until $t = T/2$
and containing ‘0’ at least once when $T/2 < t \leq T$

Recursion

I are the sequences containing '0' at least once before $t = T/2$

II are the sequences containing only '1's until $t = T/2$ and containing '0' at least once when $T/2 < t \leq T$

$$\mathbb{P}\left(\text{I}\right) + \mathbb{P}\left(\text{II}\right) \leq \frac{3}{2} q_0 + \delta \qquad \Longrightarrow$$

Recursion

I are the sequences containing '0' at least once before $t = T/2$

II are the sequences containing only '1's until $t = T/2$ and containing '0' at least once when $T/2 < t \leq T$

$$\mathbb{P}\left(\mathrm{I}\right) + \mathbb{P}\left(\mathrm{II}\right) \leq \frac{3}{2}q_0 + \delta \quad \Longrightarrow \quad \mathbb{P}\left(\mathrm{I}\right) \leq \frac{3}{4}q_0 + \frac{\delta}{2} \quad \text{or} \quad \mathbb{P}\left(\mathrm{II}\right) \leq \frac{3}{4}q_0 + \frac{\delta}{2}$$

Recursion

$\boxed{\text{I}}$ are the sequences containing '0' at least once before $t = T/2$

$\boxed{\text{II}}$ are the sequences containing only '1's until $t = T/2$ and containing '0' at least once when $T/2 < t \leq T$

$$\mathbb{P}\left(\boxed{\text{I}}\right) + \mathbb{P}\left(\boxed{\text{II}}\right) \leq \frac{3}{2}q_0 + \delta \quad \Longrightarrow \quad \mathbb{P}\left(\boxed{\text{I}}\right) \leq \frac{3}{4}q_0 + \frac{\delta}{2} \quad \text{or} \quad \mathbb{P}\left(\boxed{\text{II}}\right) \leq \frac{3}{4}q_0 + \frac{\delta}{2}$$

if $\mathbb{P}\left(\boxed{\text{I}}\right) \leq \frac{3}{4}q_0 + \frac{\delta}{2}$, reiterate in the first half and set $q_1 := \mathbb{P}\left(\boxed{\text{I}}\right)$

Recursion

$\boxed{\text{I}}$ are the sequences containing '0' at least once before $t = T/2$

$\boxed{\text{II}}$ are the sequences containing only '1's until $t = T/2$ and containing '0' at least once when $T/2 < t \leq T$

$$\mathbb{P}\left(\boxed{\text{I}}\right) + \mathbb{P}\left(\boxed{\text{II}}\right) \leq \frac{3}{2}q_0 + \delta \quad \Longrightarrow \quad \mathbb{P}\left(\boxed{\text{I}}\right) \leq \frac{3}{4}q_0 + \frac{\delta}{2} \quad \text{or} \quad \mathbb{P}\left(\boxed{\text{II}}\right) \leq \frac{3}{4}q_0 + \frac{\delta}{2}$$

if $\mathbb{P}\left(\boxed{\text{I}}\right) \leq \frac{3}{4}q_0 + \frac{\delta}{2}$, reiterate in the first half and set $q_1 := \mathbb{P}\left(\boxed{\text{I}}\right)$

if $\mathbb{P}\left(\boxed{\text{II}}\right) \leq \frac{3}{4}q_0 + \frac{\delta}{2}$, reiterate in the second half and set $q_1 := \mathbb{P}\left(\boxed{\text{II}}\right)$

Recursion

$\boxed{\mathrm{I}}$ are the sequences containing ‘0’ at least once before $t = T/2$

$\boxed{\mathrm{II}}$ are the sequences containing only ‘1’s until $t = T/2$ and containing ‘0’ at least once when $T/2 < t \leq T$

$$\mathbb{P}\left(\mathrm{I}\right) + \mathbb{P}\left(\mathrm{II}\right) \leq \frac{3}{2}q_0 + \delta \implies \mathbb{P}\left(\mathrm{I}\right) \leq \frac{3}{4}q_0 + \frac{\delta}{2} \quad \text{or} \quad \mathbb{P}\left(\mathrm{II}\right) \leq \frac{3}{4}q_0 + \frac{\delta}{2}$$

if $\mathbb{P}\left(\mathrm{I}\right) \leq \frac{3}{4}q_0 + \frac{\delta}{2}$, reiterate in the first half and set $q_1 := \mathbb{P}\left(\mathrm{I}\right)$

if $\mathbb{P}\left(\mathrm{II}\right) \leq \frac{3}{4}q_0 + \frac{\delta}{2}$, reiterate in the second half and set $q_1 := \mathbb{P}\left(\mathrm{II}\right)$

If we continue, we get $q_{i+1} \leq \frac{3}{4}q_i + \frac{\delta}{2} \implies q_i \leq 2\delta$

Constructing hard sequence

Constructing hard sequence

$\mathbb{P}(\mathbb{A}(s^{(k)})$ outputs a 0 in the k^{th} blue blocks$) = q_k \leq 2\delta$

Constructing hard sequence

$\mathbb{P}(\mathbb{A}(s^{(k)})$ outputs a 0 in the k^{th} blue blocks$) = q_k \leq 2\delta$

($s^{(k)}$ and $s^{(k+1)}$ are neighbouring datasets)

Constructing hard sequence

$\mathbb{P}(\mathbb{A}(s^{(k)})$ outputs a 0 in the k^{th} blue blocks$) = q_k \leq 2\delta$

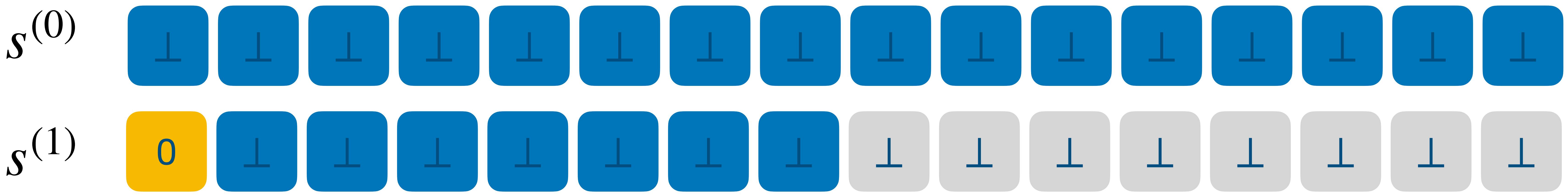
$(s^{(k)}$ and $s^{(k+1)}$ are neighbouring datasets$)$

$s^{(0)}$

Constructing hard sequence

$\mathbb{P}(\mathbb{A}(s^{(k)})$ outputs a 0 in the k^{th} blue blocks$) = q_k \leq 2\delta$

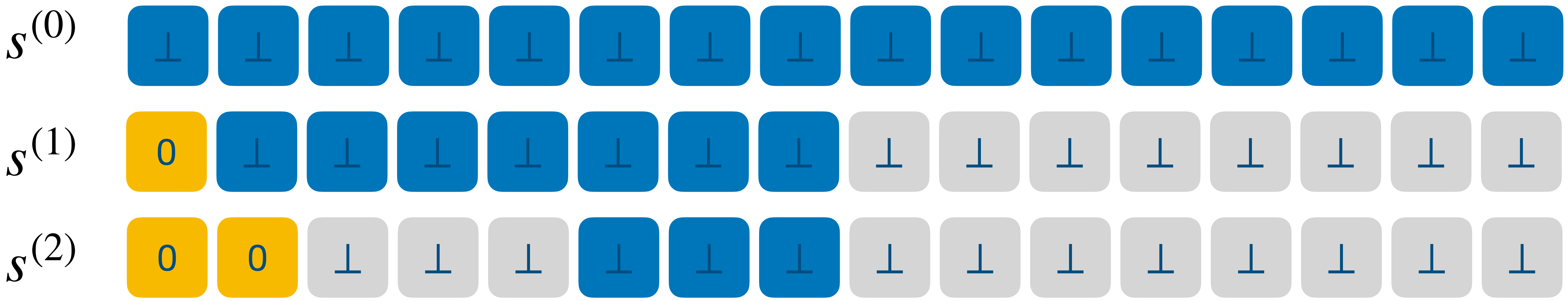
($s^{(k)}$ and $s^{(k+1)}$ are neighbouring datasets)



Constructing hard sequence

$\mathbb{P}(\mathbb{A}(s^{(k)})$ outputs a 0 in the k^{th} blue blocks$) = q_k \leq 2\delta$

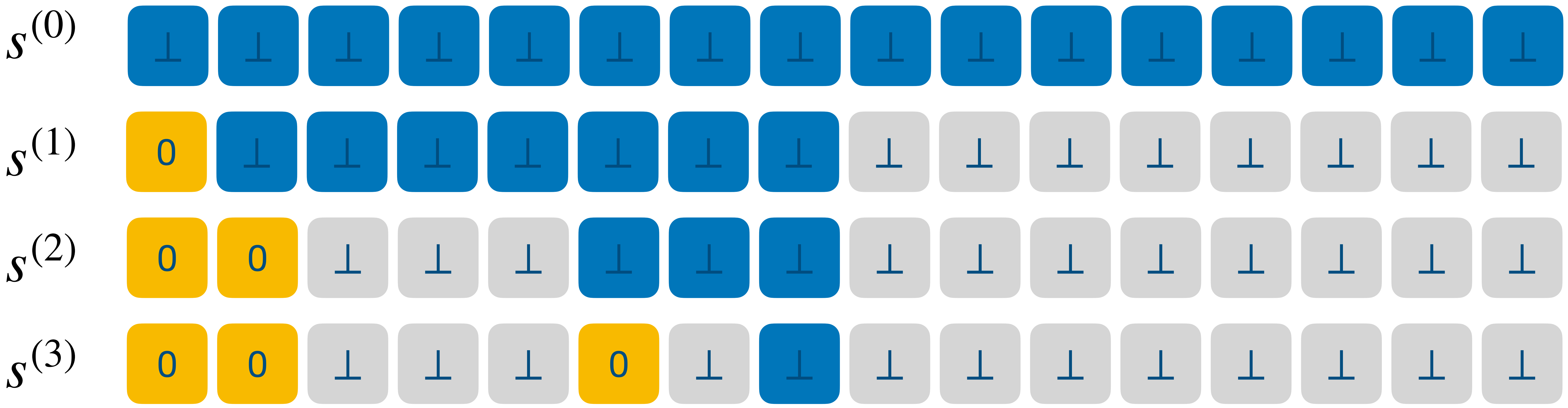
$(s^{(k)}$ and $s^{(k+1)}$ are neighbouring datasets$)$



Constructing hard sequence

$\mathbb{P}(\mathbb{A}(s^{(k)})$ outputs a 0 in the k^{th} blue blocks$) = q_k \leq 2\delta$

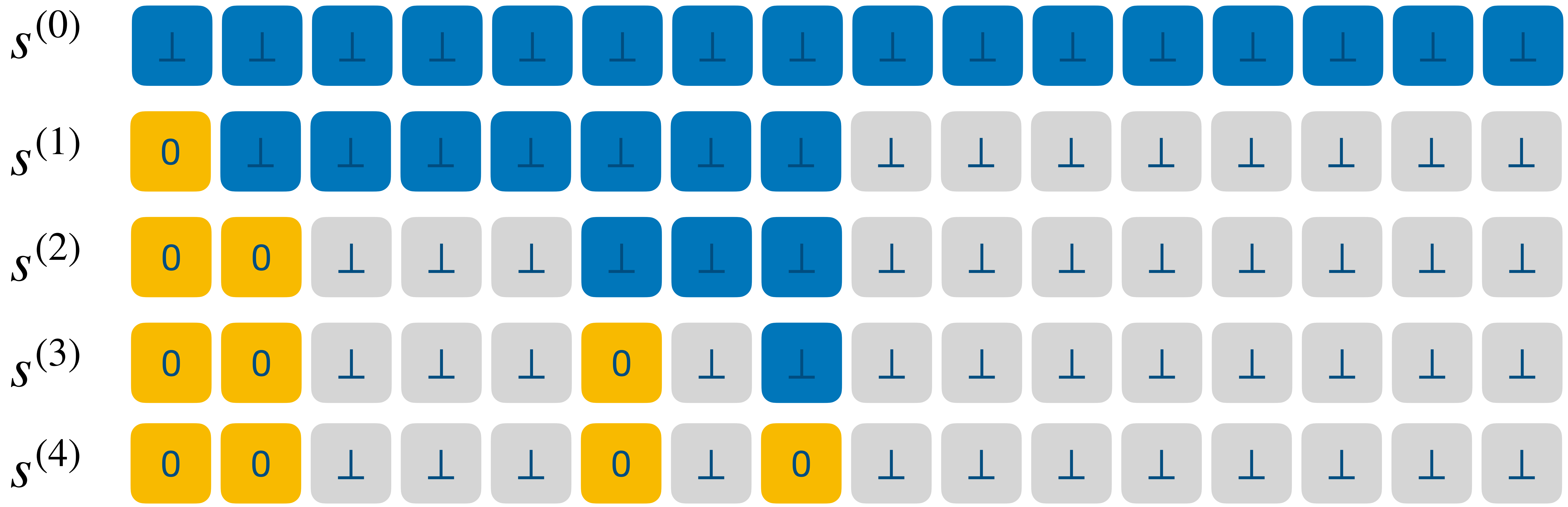
$(s^{(k)}$ and $s^{(k+1)}$ are neighbouring datasets)



Constructing hard sequence

$\mathbb{P}(\mathbb{A}(s^{(k)})$ outputs a 0 in the k^{th} blue blocks$) = q_k \leq 2\delta$

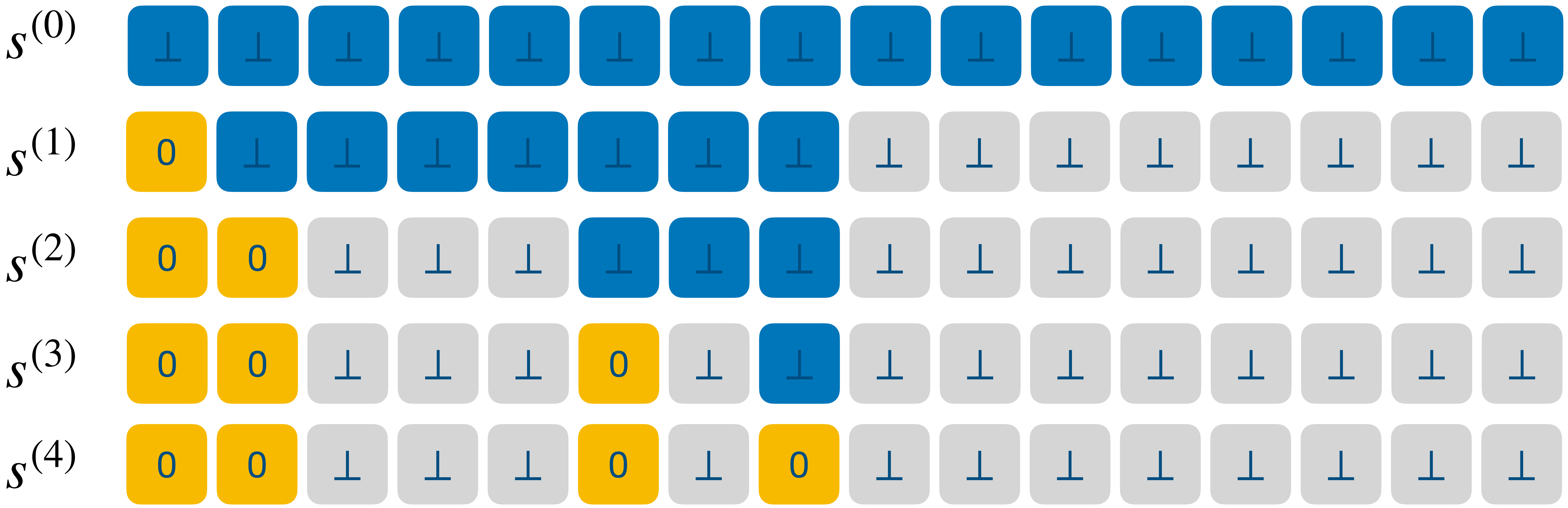
($s^{(k)}$ and $s^{(k+1)}$ are neighbouring datasets)



Constructing hard sequence

$\mathbb{P}(\mathbb{A}(s^{(k)})$ outputs a 0 in the k^{th} blue blocks$) = q_k \leq 2\delta$

($s^{(k)}$ and $s^{(k+1)}$ are neighbouring datasets)

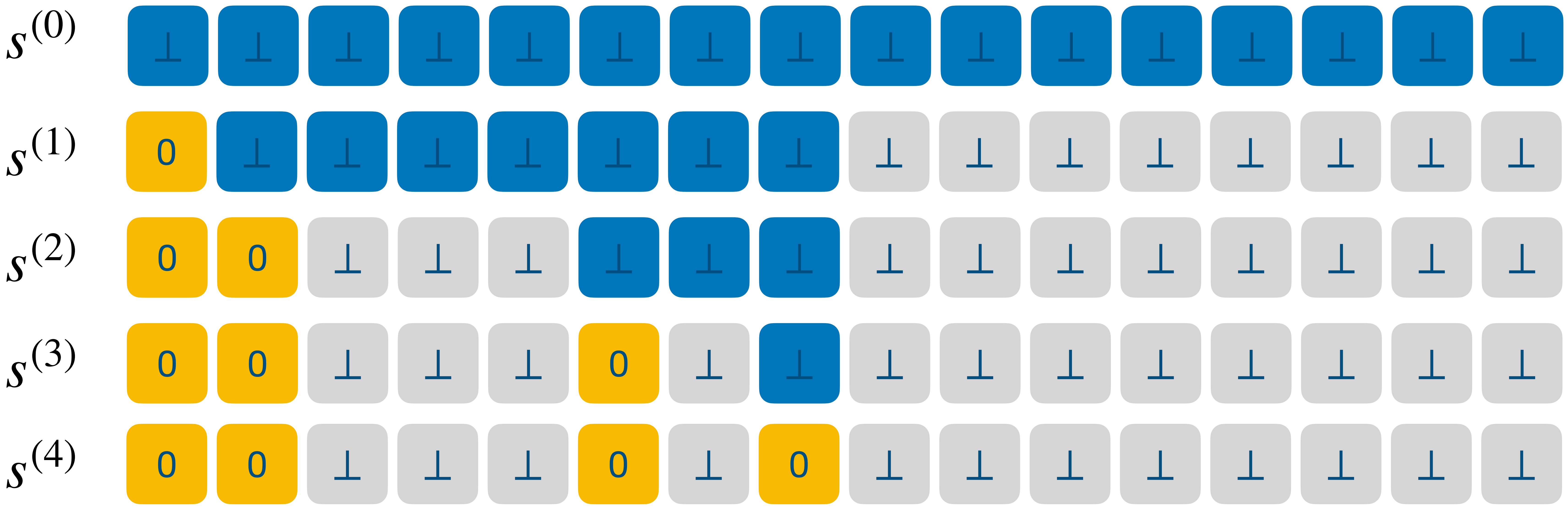


$\mathbb{P}(\mathbb{A}(s^{(k)})$ makes less than k mistakes$)$

Constructing hard sequence

$\mathbb{P}(\mathbb{A}(s^{(k)})$ outputs a 0 in the k^{th} blue blocks$) = q_k \leq 2\delta$

($s^{(k)}$ and $s^{(k+1)}$ are neighbouring datasets)



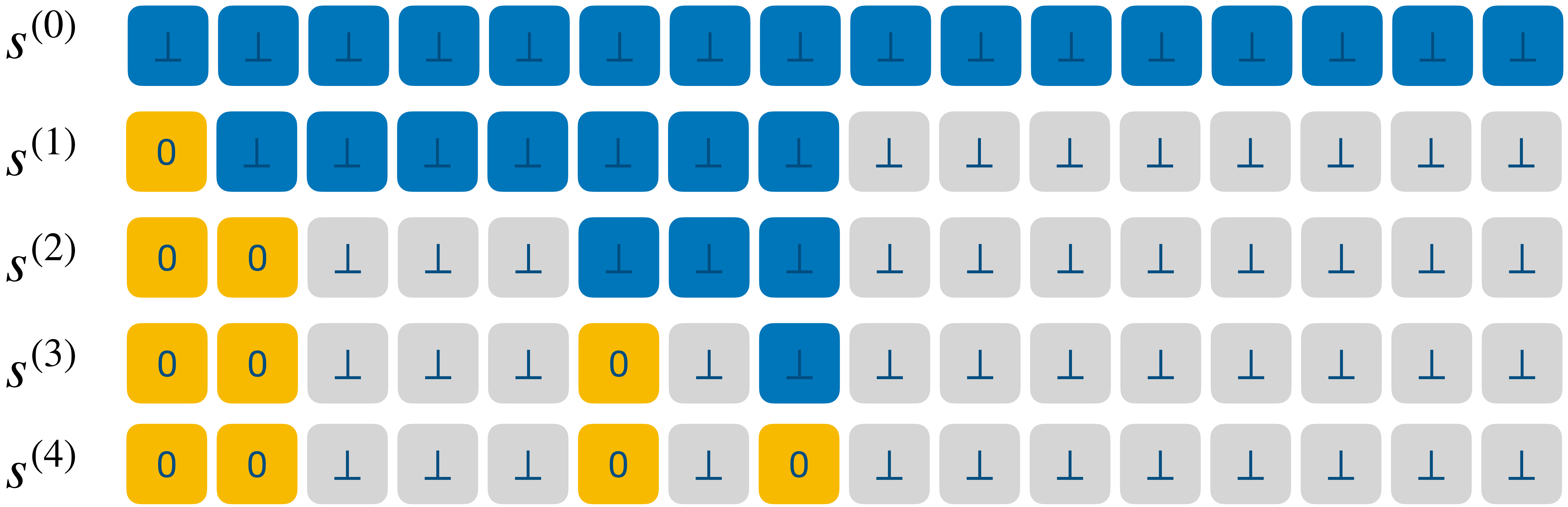
$\mathbb{P}(\mathbb{A}(s^{(k)})$ makes less than k mistakes$)$

$\leq \mathbb{P}(\exists i^{\text{th}}$ blue blocks, such that $\mathbb{A}(s^{(i)})$ contains $0)$

Constructing hard sequence

$\mathbb{P}(\mathbb{A}(s^{(k)})$ outputs a 0 in the k^{th} blue blocks$) = q_k \leq 2\delta$

$(s^{(k)}$ and $s^{(k+1)}$ are neighbouring datasets)



$\mathbb{P}(\mathbb{A}(s^{(k)})$ makes less than k mistakes)

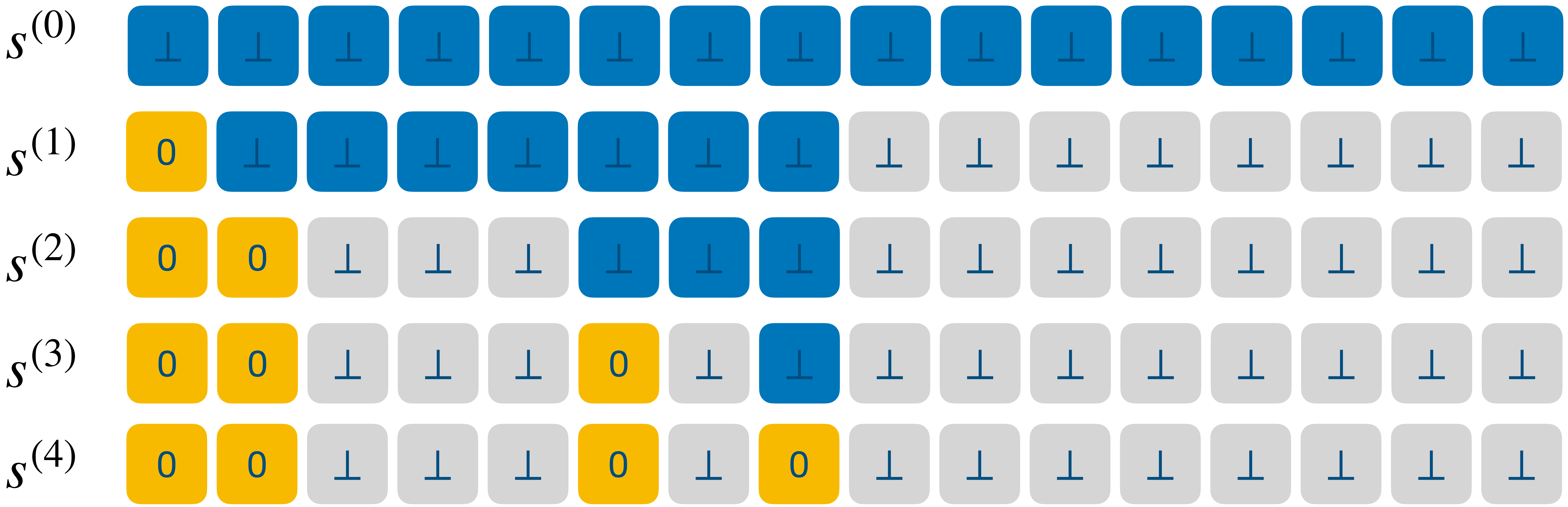
$\leq \mathbb{P}(\exists i^{\text{th}}$ blue blocks, such that $\mathbb{A}(s^{(i)})$ contains $0)$

$$\leq \sum_{i=1}^{k} q_i \leq 2k\delta$$

Constructing hard sequence

$\mathbb{P}(\mathbb{A}(s^{(k)})$ outputs a 0 in the k^{th} blue blocks$) = q_k \leq 2\delta$

($s^{(k)}$ and $s^{(k+1)}$ are neighbouring datasets)



$\mathbb{P}(\mathbb{A}(s^{(k)})$ makes less than k mistakes$)$

$\leq \mathbb{P}(\exists i^{\text{th}}$ blue blocks, such that $\mathbb{A}(s^{(i)})$ contains $0)$

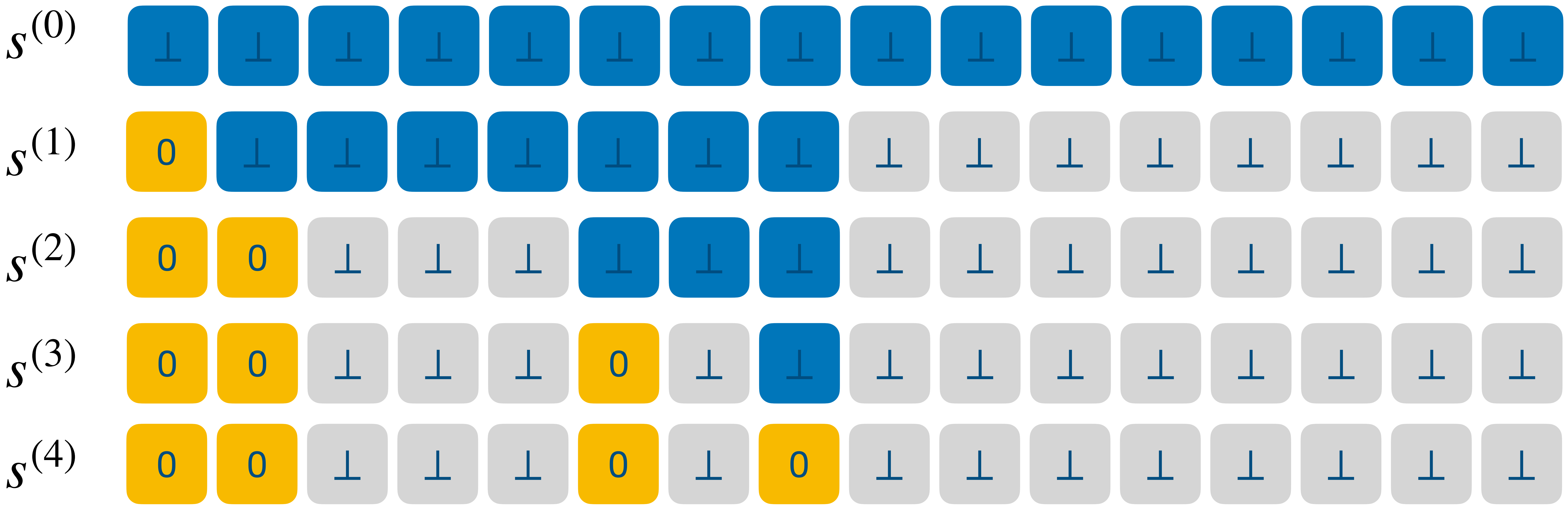
$$\leq \sum_{i=1}^{k} q_i \leq 2k\delta$$

$\Longrightarrow \mathbb{P}(\mathbb{A}(s^{(k)})$ makes k mistakes$) \geq 1/2,$

Constructing hard sequence

$\mathbb{P}(\mathbb{A}(s^{(k)})$ outputs a 0 in the k^{th} blue blocks$) = q_k \leq 2\delta$

($s^{(k)}$ and $s^{(k+1)}$ are neighbouring datasets)



$\mathbb{P}(\mathbb{A}(s^{(k)})$ makes less than k mistakes$)$

$\leq \mathbb{P}(\exists i^{\text{th}}$ blue blocks, such that $\mathbb{A}(s^{(i)})$ contains $0)$

$$\leq \sum_{i=1}^{k} q_i \leq 2k\delta$$

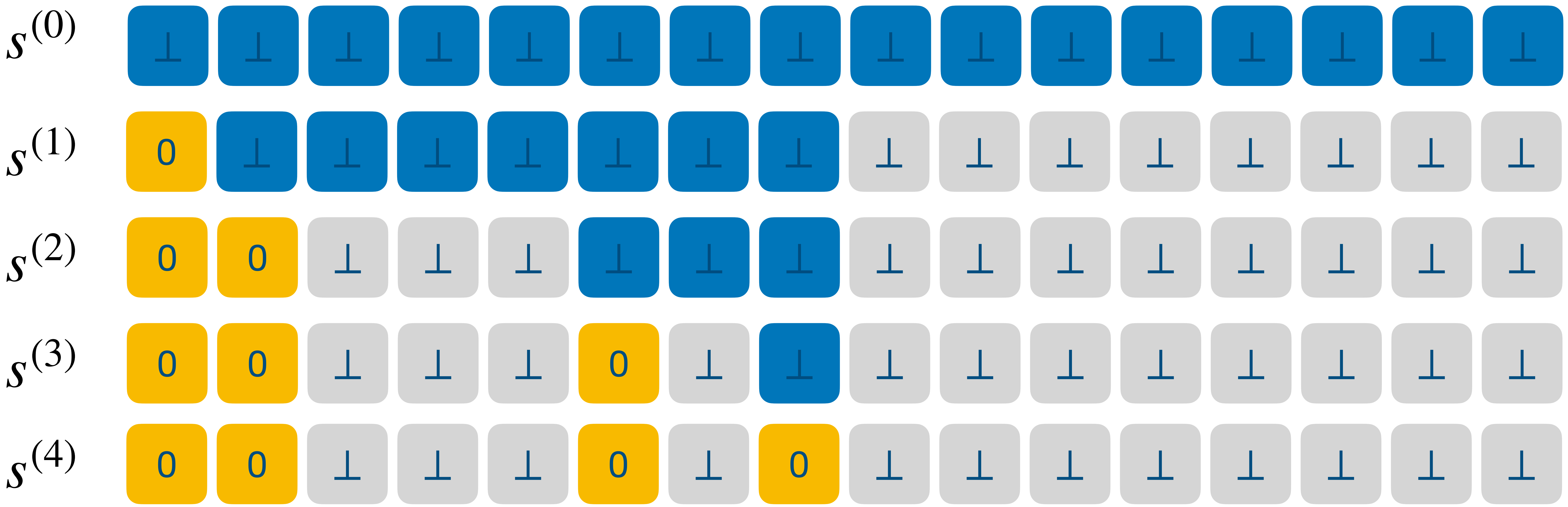
$\implies \mathbb{P}(\mathbb{A}(s^{(k)})$ makes k mistakes$) \geq 1/2$,

As long as $k \leq \frac{1}{2} \log T \leq \frac{1}{4\delta}$

Constructing hard sequence

$\mathbb{P}(\mathbb{A}(s^{(k)})$ outputs a 0 in the k^{th} blue blocks$) = q_k \leq 2\delta$

$(s^{(k)}$ and $s^{(k+1)}$ are neighbouring datasets$)$



$\mathbb{P}(\mathbb{A}(s^{(k)})$ makes less than k mistakes$)$

$\leq \mathbb{P}(\exists i^{\text{th}}$ blue blocks, such that $\mathbb{A}(s^{(i)})$ contains $0)$

$$\leq \sum_{i=1}^{k} q_i \leq 2k\delta$$

$\implies \mathbb{P}(\mathbb{A}(s^{(k)})$ makes k mistakes$) \geq 1/2$,

As long as $k \leq \frac{1}{2}\log T \leq \frac{1}{4\delta}$

$$\implies \mathbb{E}\,[\mathrm{M}] \geq \min\left(\frac{\log T}{4}, \frac{1}{4\delta}\right)$$

Takeaways and open problems

Takeaways and open problems

Lower bounds

Takeaways and open problems

Lower bounds

	$\mathbb{X} = \{0,1\}$ $F = \{f^{(0)}, f^{(1)}\}$	

Takeaways and open problems

Lower bounds

	$\mathbb{X} = \{0,1\}$ $F = \{f^{(0)}, f^{(1)}\}$	Point_∞

Takeaways and open problems

Lower bounds

	$\mathbb{X} = \{0,1\}$ $F = \{f^{(0)}, f^{(1)}\}$	Point_∞

Our contribution
for concentrated (and uniform) algorithms

Takeaways and open problems

Lower bounds

	$\mathbb{X} = \{0,1\}$ $F = \{f^{(0)}, f^{(1)}\}$	Point_∞
Pure DP		

Our contribution
for concentrated (and uniform) algorithms

Takeaways and open problems

Lower bounds

	$\mathbb{X} = \{0,1\}$ $F = \{f^{(0)}, f^{(1)}\}$	Point_∞
Pure DP	$\log T$	

Our contribution
for concentrated (and uniform) algorithms

Takeaways and open problems

Lower bounds

	$\mathbb{X} = \{0,1\}$ $F = \{f^{(0)}, f^{(1)}\}$	Point_∞
Pure DP	$\log T$	$\log T$

Our contribution
for concentrated (and uniform) algorithms

Takeaways and open problems

Lower bounds

	$\mathbb{X} = \{0,1\}$ $F = \{f^{(0)}, f^{(1)}\}$	Point_∞
Pure DP	$\log T$	$\log T$
Approximate DP		

Our contribution
for concentrated (and uniform) algorithms

Takeaways and open problems

Lower bounds

	$\mathbb{X} = \{0,1\}$ $F = \{f^{(0)}, f^{(1)}\}$	Point_∞
Pure DP	$\log T$	$\log T$
Approximate DP	$\min(\log T,\ 1/\delta)$	

Our contribution
for concentrated (and uniform) algorithms

Takeaways and open problems

Lower bounds

	$\mathbb{X} = \{0,1\}$ $F = \{f^{(0)}, f^{(1)}\}$	Point_∞
Pure DP	$\log T$	$\log T$
Approximate DP	$\min(\log T,\ 1/\delta)$	$\min(\log T,\ 1/\delta)$

Our contribution
for concentrated (and uniform) algorithms

Takeaways and open problems

Lower bounds

	$\mathbb{X} = \{0,1\}$ $F = \{f^{(0)}, f^{(1)}\}$	Point_∞
Pure DP	$\log T$	$\log T$
Approximate DP	$\min(\log T,\ 1/\delta)$	$\min(\log T,\ 1/\delta)$

Our contribution
for concentrated (and uniform) algorithms

For any algorithm
[CLNSS24]

Takeaways and open problems

Lower bounds

	$\mathbb{X} = \{0,1\}$ $F = \{f^{(0)}, f^{(1)}\}$	Point_∞
Pure DP	$\log T$	$\log T$
Approximate DP	$\min(\log T,\ 1/\delta)$	$\min(\log T,\ 1/\delta)$

Our contribution
for concentrated (and uniform) algorithms

For any algorithm
[CLNSS24]

Takeaways and open problems

Conjecture
(For any algorithm)

Lower bounds

	$\mathbb{X} = \{0,1\}$ $F = \{f^{(0)}, f^{(1)}\}$	Point_∞
Pure DP	$\log T$	$\log T$
Approximate DP	$\min(\log T,\ 1/\delta)$	$\min(\log T,\ 1/\delta)$

Our contribution
for concentrated (and uniform) algorithms

For any algorithm
[CLNSS24]

Takeaways and open problems

Conjecture
(For any algorithm)

Upper bounds

	$\mathbb{X} = \{0,1\}$, $F = \{f^{(0)}, f^{(1)}\}$	

Lower bounds

	$\mathbb{X} = \{0,1\}$, $F = \{f^{(0)}, f^{(1)}\}$	Point_∞
Pure DP	$\log T$	$\log T$
Approximate DP	$\min(\log T,\ 1/\delta)$	$\min(\log T,\ 1/\delta)$

Our contribution
for concentrated (and uniform) algorithms

For any algorithm
[CLNSS24]

Takeaways and open problems

Conjecture
(For any algorithm)

Upper bounds

	$\mathbb{X} = \{0,1\}$ $F = \{f^{(0)}, f^{(1)}\}$	Point_∞

Lower bounds

	$\mathbb{X} = \{0,1\}$ $F = \{f^{(0)}, f^{(1)}\}$	Point_∞
Pure DP	$\log T$	$\log T$
Approximate DP	$\min(\log T,\ 1/\delta)$	$\min(\log T,\ 1/\delta)$

Our contribution
for concentrated (and uniform) algorithms

For any algorithm
[CLNSS24]

Takeaways and open problems

Conjecture
(For any algorithm)

Upper bounds

	$\mathbb{X} = \{0,1\}$ $F = \{f^{(0)}, f^{(1)}\}$	Point_∞
Pure DP		

Lower bounds

	$\mathbb{X} = \{0,1\}$ $F = \{f^{(0)}, f^{(1)}\}$	Point_∞
Pure DP	$\log T$	$\log T$
Approximate DP	$\min(\log T,\ 1/\delta)$	$\min(\log T,\ 1/\delta)$

Our contribution
for concentrated (and uniform) algorithms

For any algorithm
[CLNSS24]

Takeaways and open problems

Conjecture (For any algorithm)

Upper bounds

	$\mathbb{X} = \{0,1\}$ $F = \{f^{(0)}, f^{(1)}\}$	Point_∞
Pure DP	$\text{poly} \log T$	

Lower bounds

	$\mathbb{X} = \{0,1\}$ $F = \{f^{(0)}, f^{(1)}\}$	Point_∞
Pure DP	$\log T$	$\log T$
Approximate DP	$\min(\log T,\ 1/\delta)$	$\min(\log T,\ 1/\delta)$

Our contribution for concentrated (and uniform) algorithms

For any algorithm [CLNSS24]

Takeaways and open problems

Conjecture
(For any algorithm)

Upper bounds

	$\mathbb{X} = \{0,1\}$ $F = \{f^{(0)}, f^{(1)}\}$	Point_∞
Pure DP	$\text{poly} \log T$	∞

Lower bounds

	$\mathbb{X} = \{0,1\}$ $F = \{f^{(0)}, f^{(1)}\}$	Point_∞
Pure DP	$\log T$	$\log T$
Approximate DP	$\min(\log T,\ 1/\delta)$	$\min(\log T,\ 1/\delta)$

Our contribution
for concentrated (and uniform) algorithms

For any algorithm
[CLNSS24]

Takeaways and open problems

Conjecture (For any algorithm)

Upper bounds

	$\mathbb{X} = \{0,1\}$ $F = \{f^{(0)}, f^{(1)}\}$	Point_∞
Pure DP	$\text{poly} \log T$	∞

Lower bounds

	$\mathbb{X} = \{0,1\}$ $F = \{f^{(0)}, f^{(1)}\}$	Point_∞
Pure DP	$\log T$	$\log T$
Approximate DP	$\min(\log T,\ 1/\delta)$	$\min(\log T,\ 1/\delta)$

Via Reduction
To DP continual observation

Our contribution
for concentrated (and uniform) algorithms

For any algorithm
[CLNSS24]

Takeaways and open problems

Conjecture (For any algorithm)

Upper bounds

	$\mathbb{X} = \{0,1\}$, $F = \{f^{(0)}, f^{(1)}\}$	Point_∞
Pure DP	$\text{poly} \log T$	∞
Approximate DP	$\min(\log T,\ 1/\delta)$	$\min(\log T,\ 1/\delta)$

Lower bounds

	$\mathbb{X} = \{0,1\}$, $F = \{f^{(0)}, f^{(1)}\}$	Point_∞
Pure DP	$\log T$	$\log T$
Approximate DP	$\min(\log T,\ 1/\delta)$	$\min(\log T,\ 1/\delta)$

Via Reduction
To DP continual observation

Our contribution
for concentrated (and uniform) algorithms

For any algorithm
[CLNSS24]

Thank you

If you are interested to work on questions like these,

I am looking to advise

- PhD & postdocs in Copenhagen and
- UGPs in IIT Kanpur.