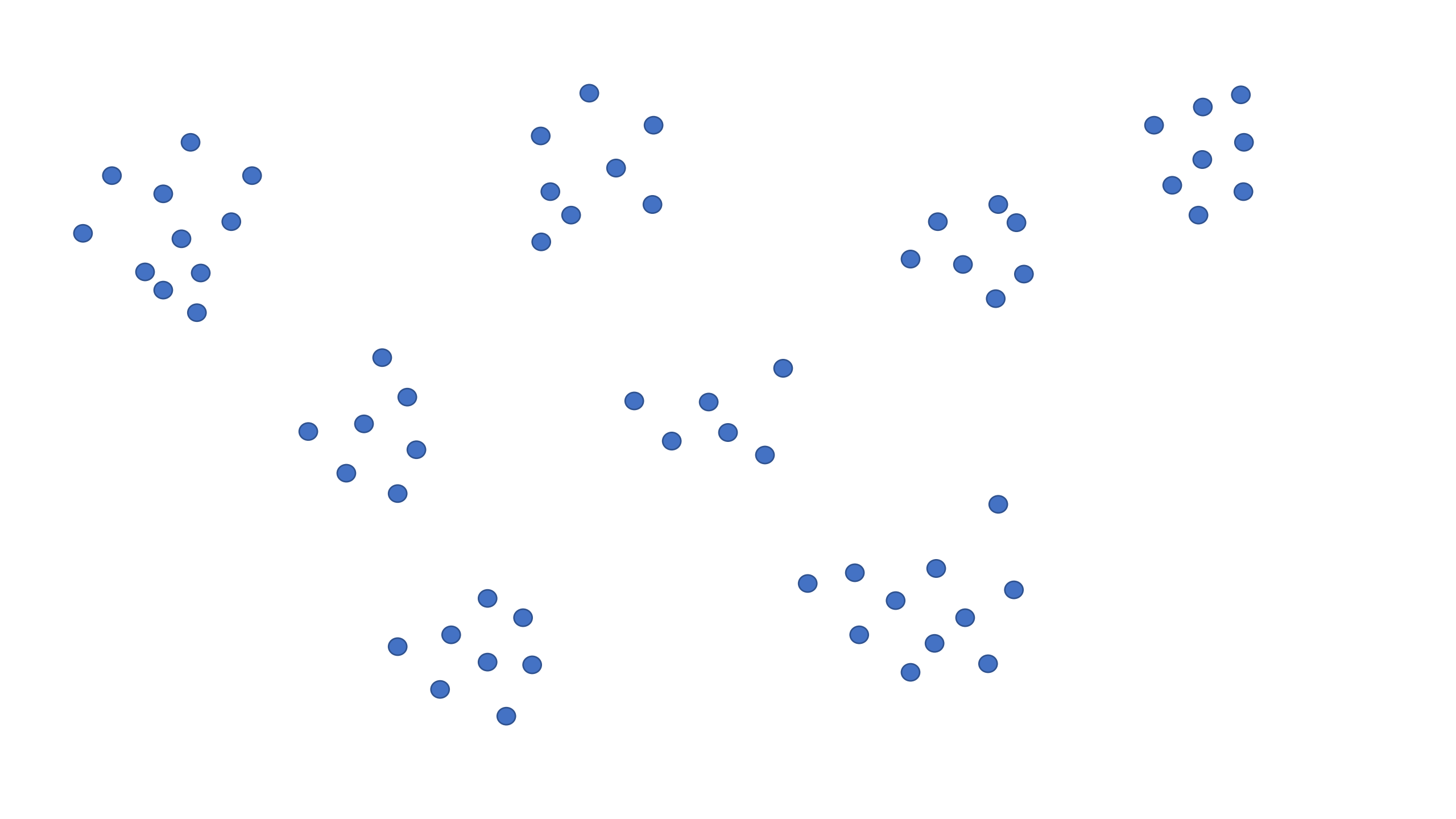
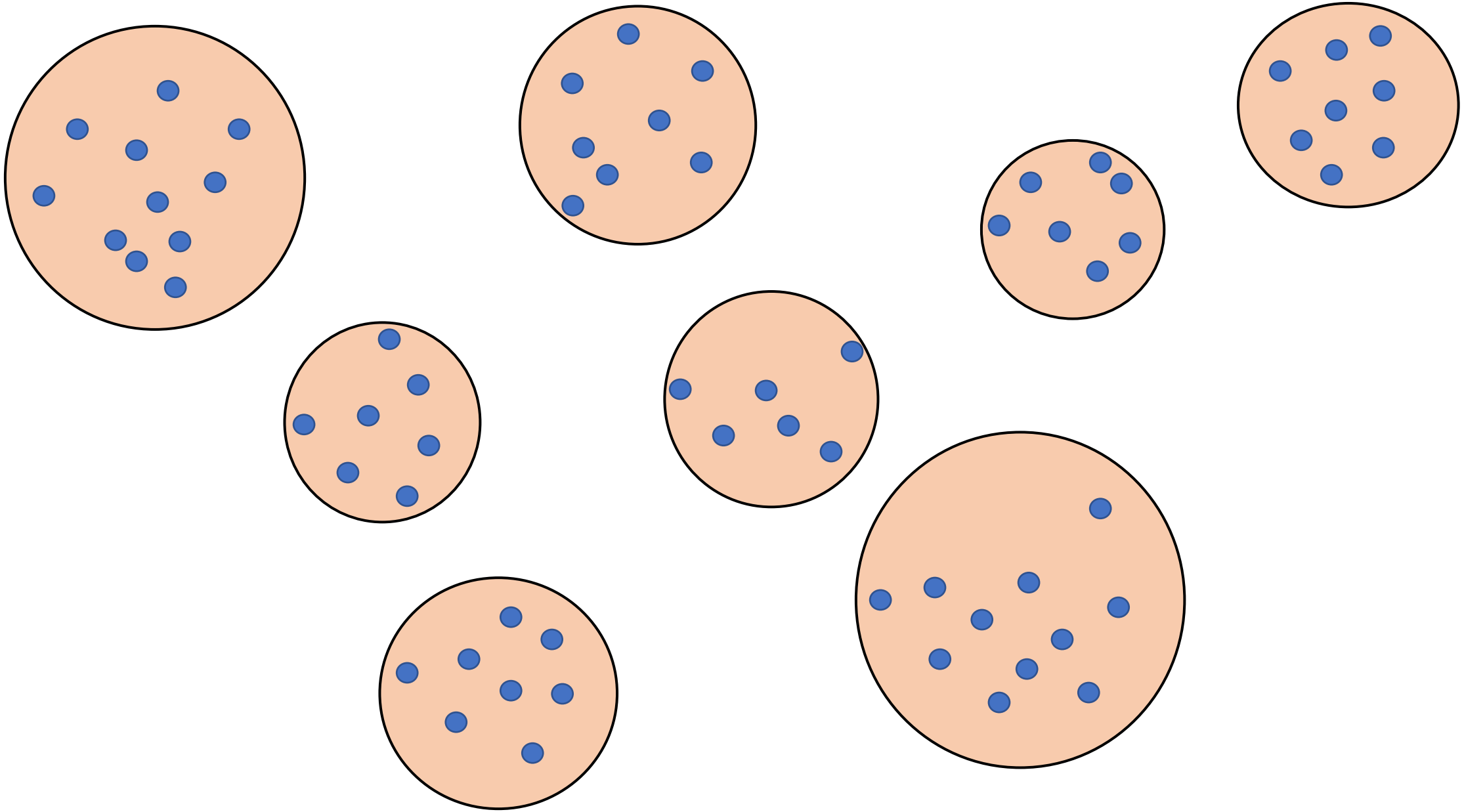
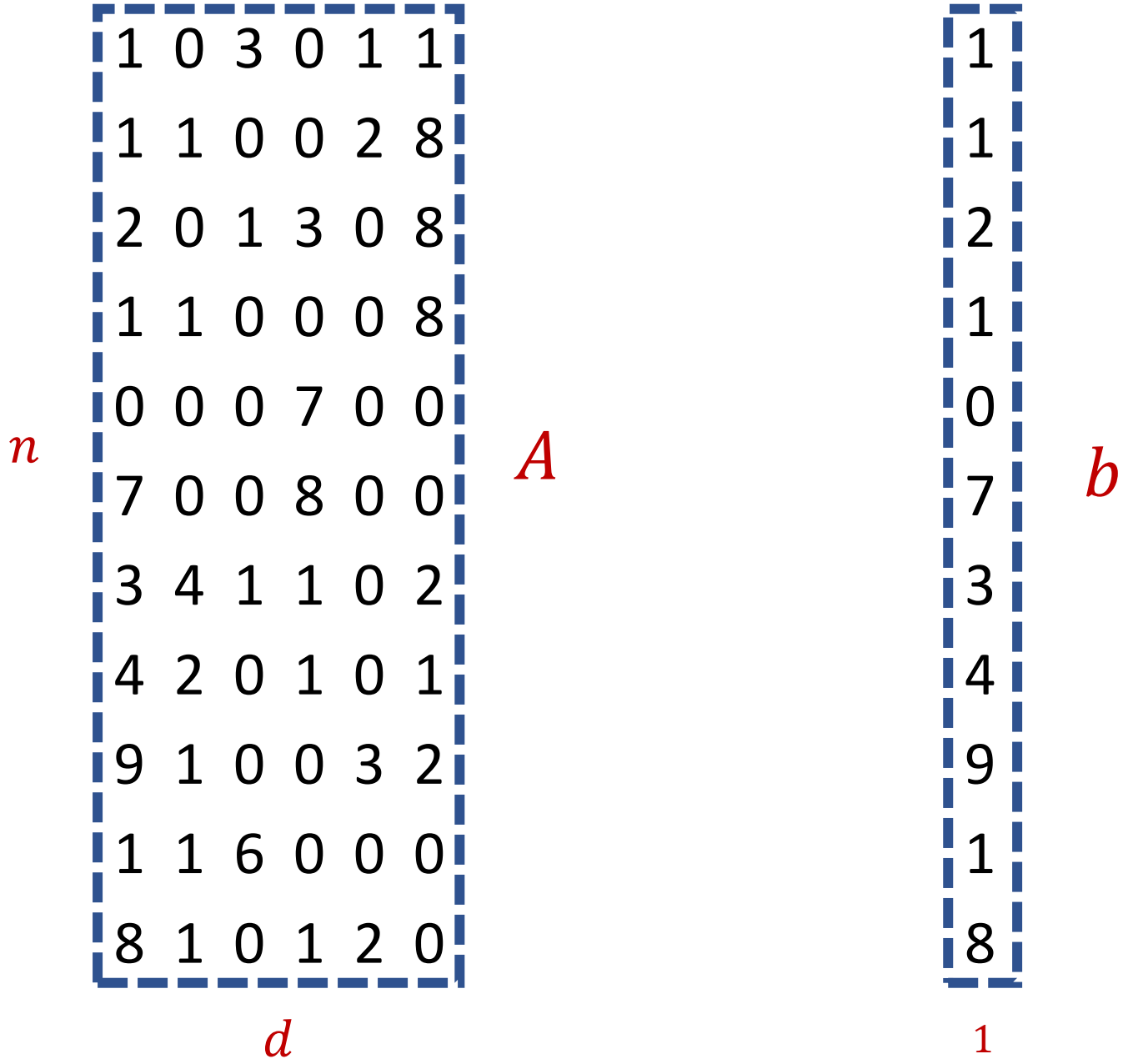
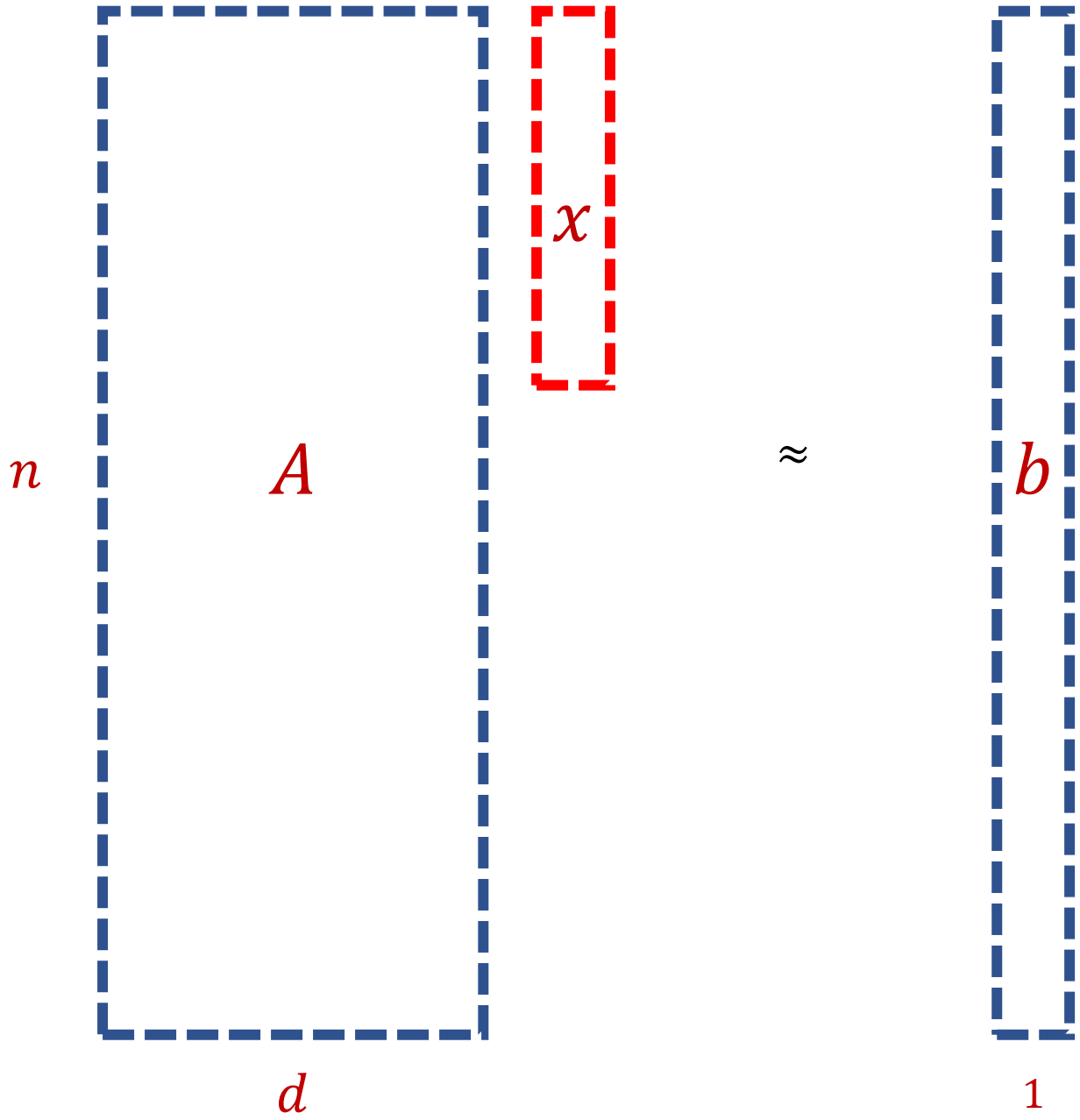
Coresets

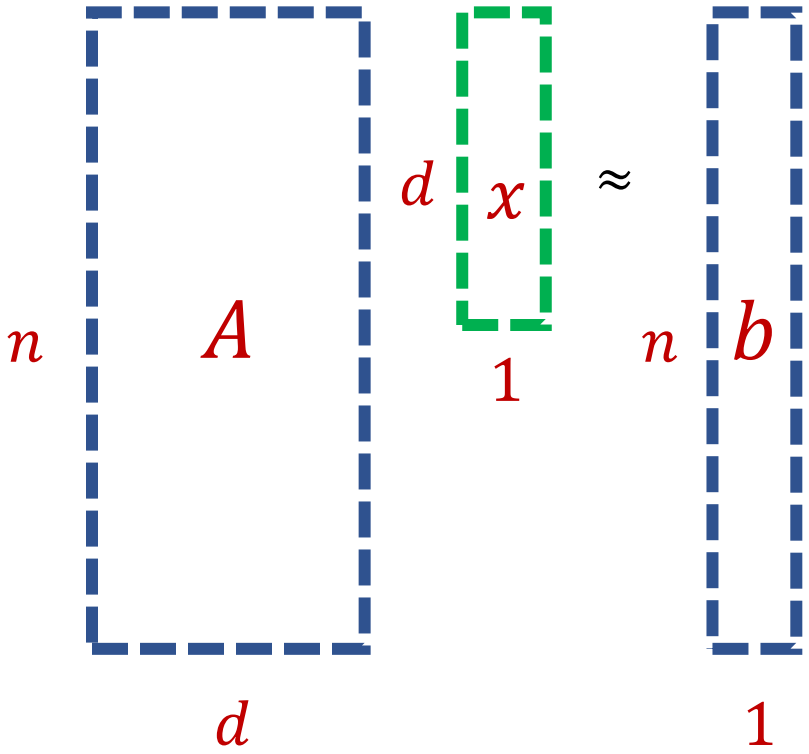





1 0 3 0 1 1
1 1 0 0 2 8
2 0 1 3 0 8
1 1 0 0 0 8
0 0 0 7 0 0
7 0 0 8 0 0
3 4 1 1 0 2
4 2 0 1 0 1
9 1 0 0 3 2
1 1 6 0 0 0
8 1 0 1 2 0
n
A
d
1
1
2
1
0
7
3
4
9
1
8
b
1

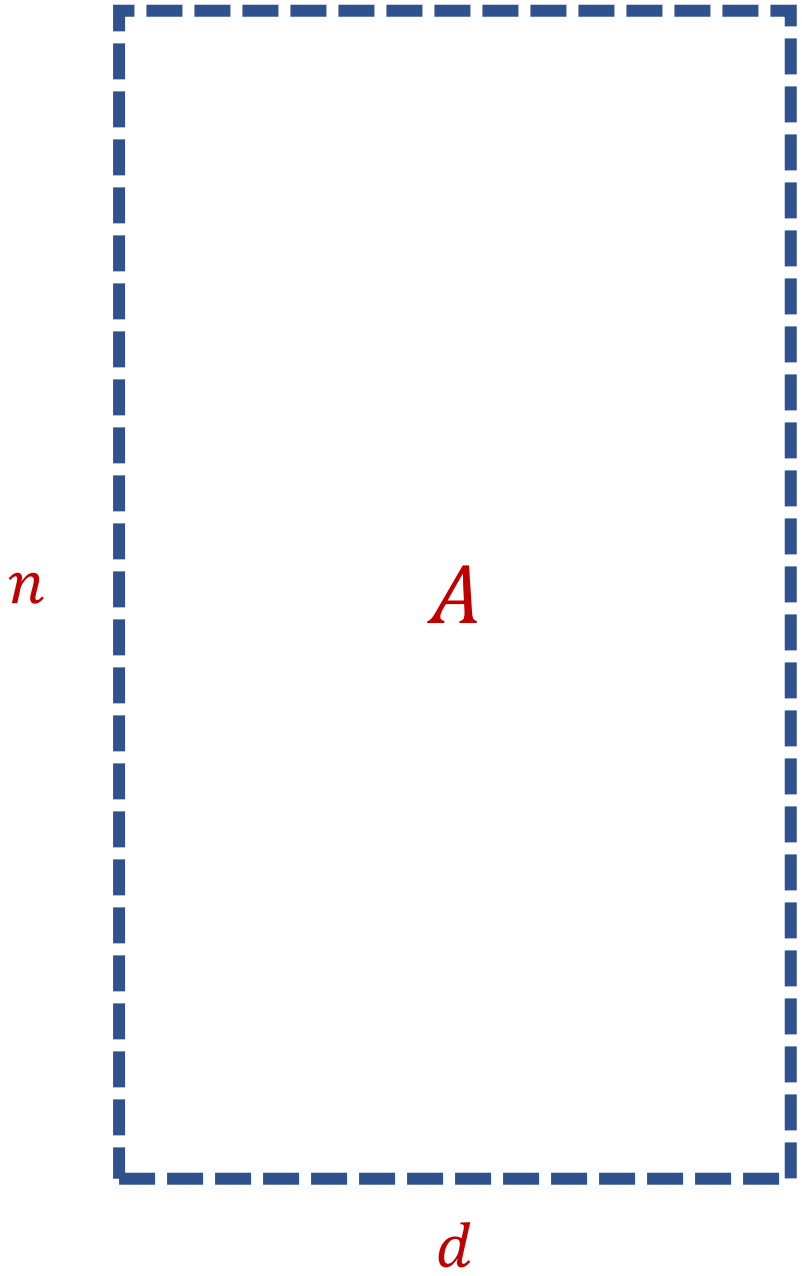

n
A
d
x
≈
b
1

Linear Regression



- Find the vector x that minimizes $\|Ax - b\|_2$
- "Least squares" optimization

- Find a vector $\hat{x}$ with $\|A\hat{x} - b\|_2 \leq (1 + \varepsilon)(\min\|Ax - b\|_2)$



Dimensionality Reduction

- Reduce d
- Reduce n

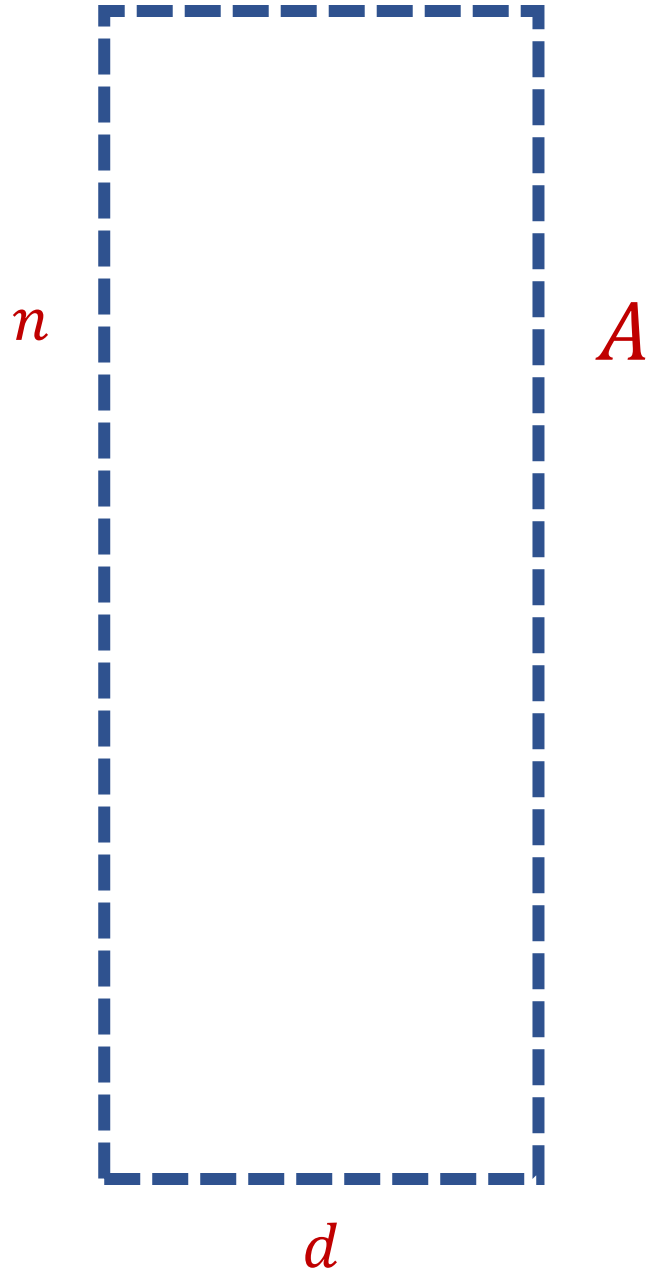
Sampling, Sketching, Johnson-Lindenstrauss, IsoMap

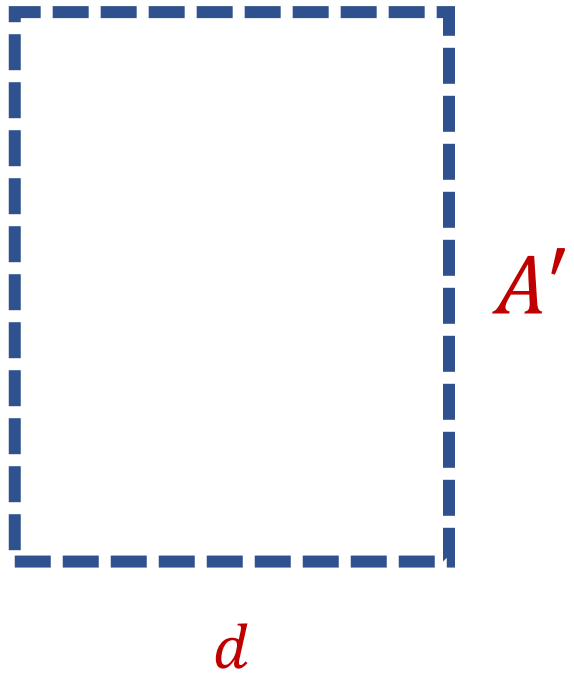
Sampling, Sketching

- Application depends on task!

Coreset

- Subset A' of representative rows of A for a given task with "score" function f
- $f(A,\cdot) \approx f(A',\cdot)$





Coreset (Formal Definition)

- Given a set X with weight function u and an accuracy parameter $\varepsilon > 0$, we say a set Y with weight function w is an $(1+\varepsilon)$-multiplicative coreset for a function f, if for all queries q in a query space Q, we have

$$(1-\varepsilon)f(Y,q,w) \leq f(X,q,u) \leq (1+\varepsilon)f(Y,q,w)$$

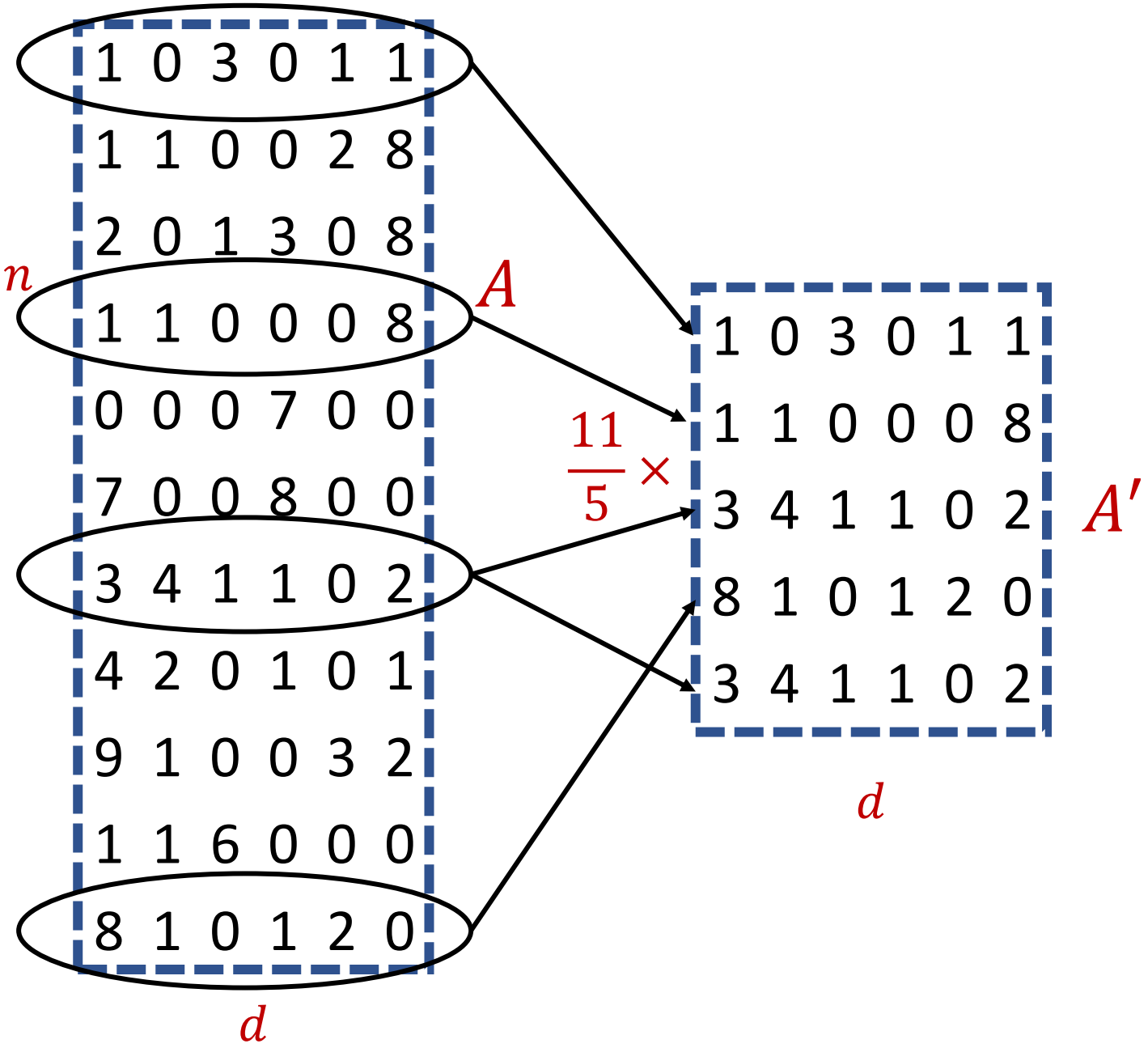
Coreset

- Lots of different constructions with various tradeoffs
  - Size vs. accuracy
  - Size vs. computation time
  - Size vs. interpretability
  - Average-case vs. worst-case performance

Uniform Sampling

- Repeat m times:
  - Pick a row A_i of A uniformly at random and scale by n/m

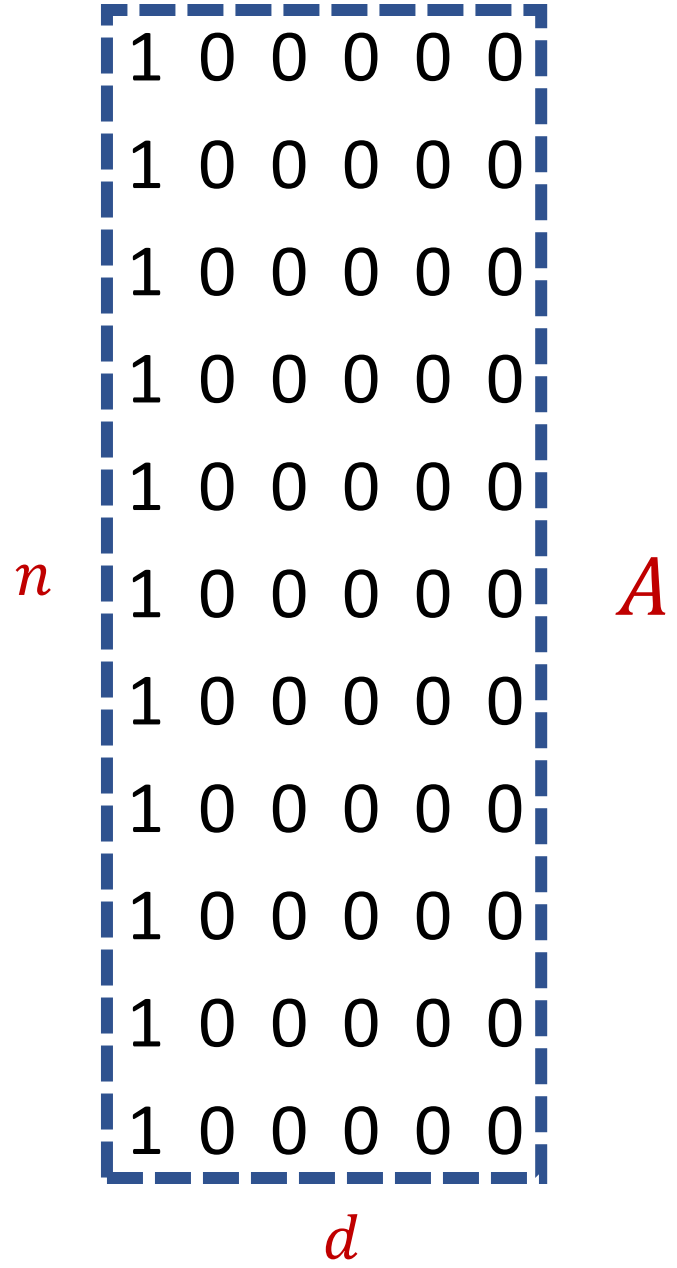
- Example with $n = 11$ and $m = 5$



Uniform Sampling

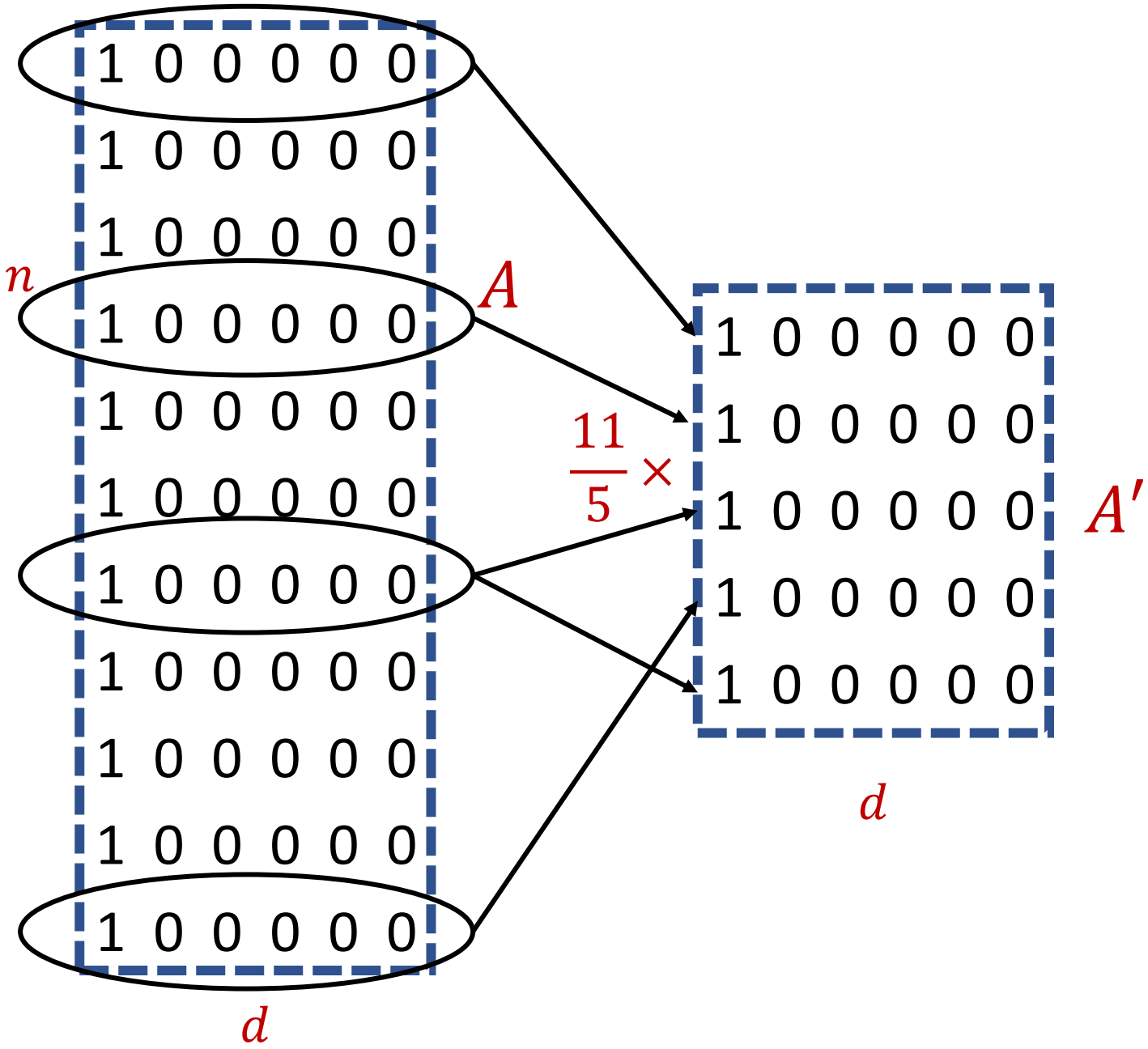
- How big should m be?

- For uniform matrix, it suffices to set $m = 1$



Uniform Sampling

- How big should m be?
- For uniform matrix, it suffices to set $m = 1$

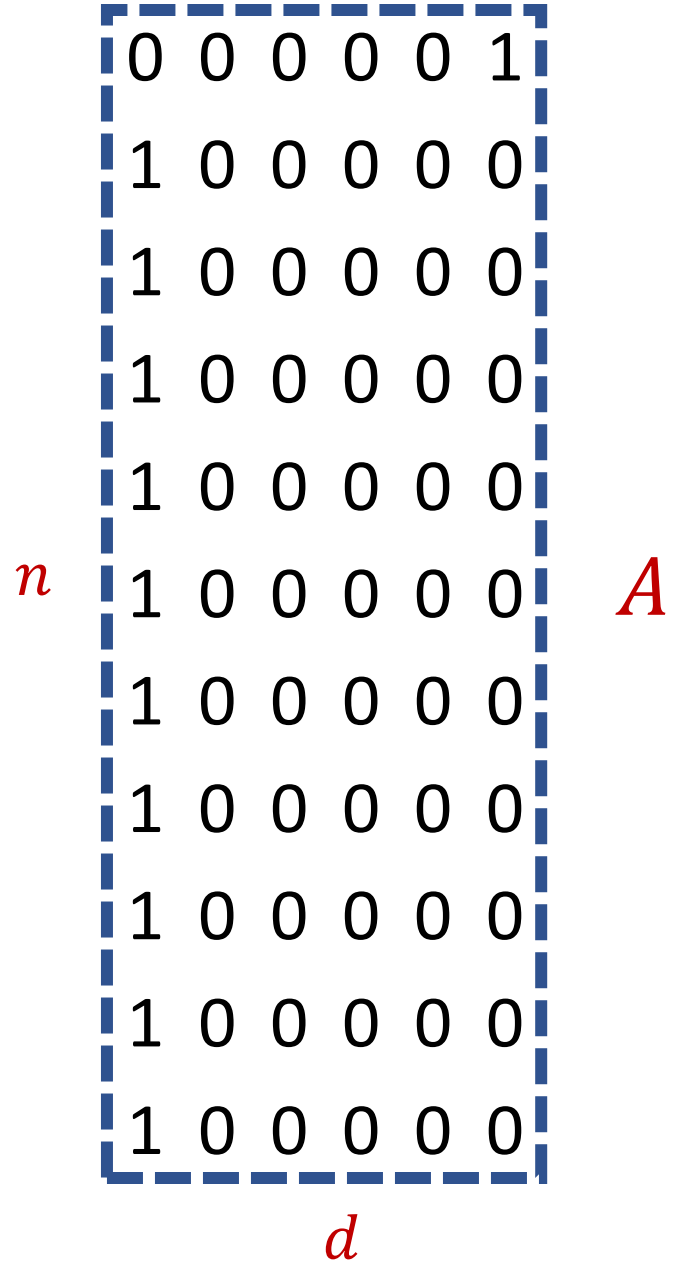


Uniform Sampling

- How big should m be?

- What is the probability the first row of A is sampled?

- Need $m = O(n \log n)$ rows



Uniform Sampling

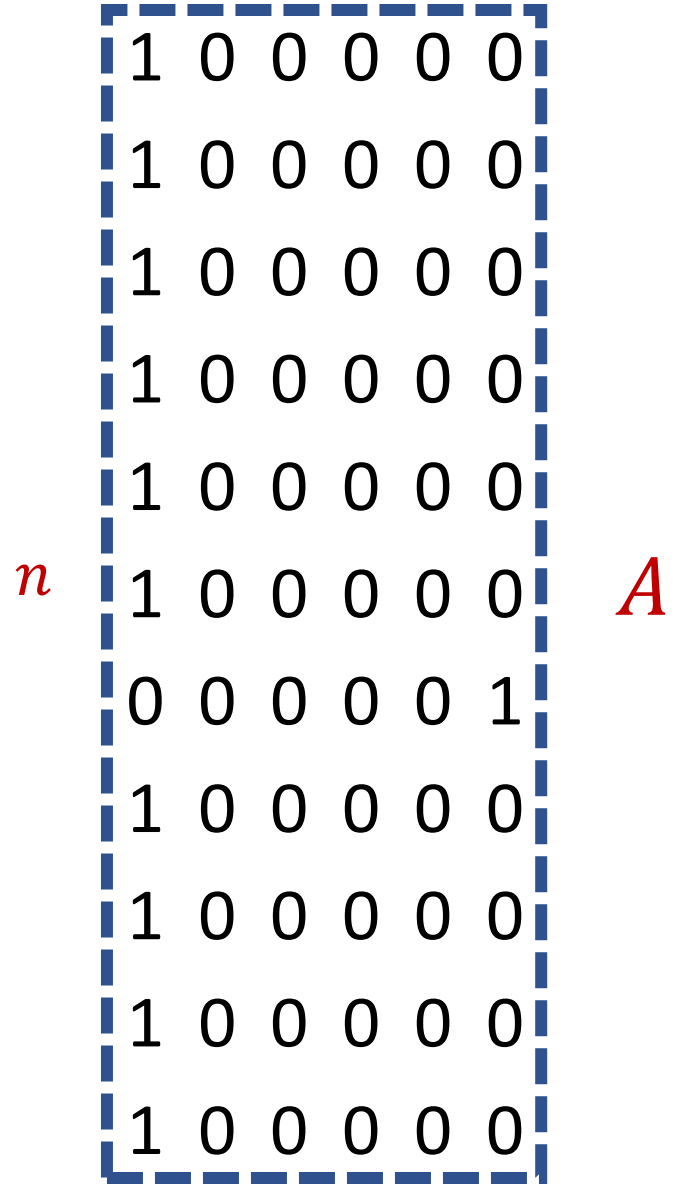
- Pros: intuitively simple, simple to implement, fast, good on uniform or “average-case” data

- Cons: bad on “worst-case” data

Importance Sampling

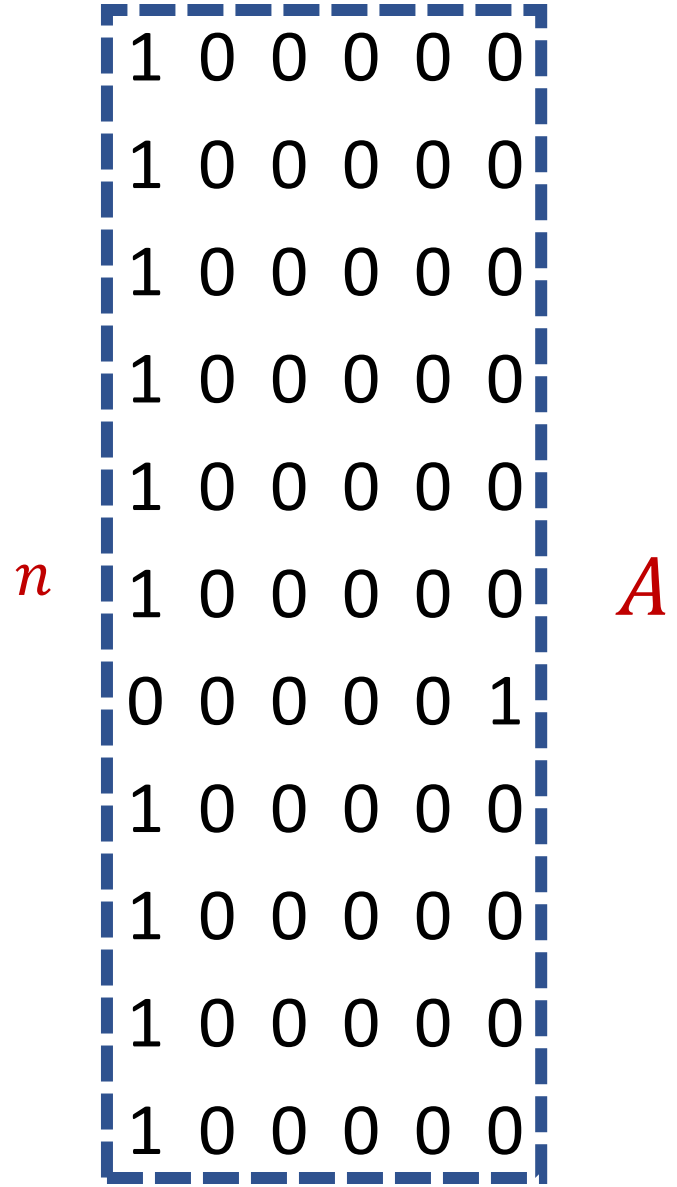
- What if we do not sample uniformly at random?

- What is the probability that each row should be sampled?



Importance Sampling

- Have a measure s_i of "importance" for each row A_i



Importance Sampling

- Have a measure s_i of "importance" for each row A_i

n | 1 0 0 0 0 0 |
| 0 0 0 0 0 1 | A

- If there were just two different rows, each row should be "maximally" important, $s_1 = s_2 = 1$

n | 1 0 0 0 0 0 |
| 1 0 0 0 0 0 |
| 1 0 0 0 0 0 |
| 1 0 0 0 0 0 |
| 1 0 0 0 0 0 |
| 1 0 0 0 0 0 |
| 1 0 0 0 0 0 |
| 0 0 0 0 0 1 |
| 1 0 0 0 0 0 |
| 1 0 0 0 0 0 |
| 1 0 0 0 0 0 |
| 1 0 0 0 0 0 | A

Importance Sampling

- Have a measure s_i of "importance" for each row A_i

- Say $s_7 = 1$ and $s_i = \frac{1}{10}$ for $i \neq 7$

$$
\begin{matrix} \\ \\ \\ \\ \\ \\ n \\ \\ \\ \\ \\ \\ \\ \end{matrix}
\begin{bmatrix}
1 & 0 & 0 & 0 & 0 & 0 \\
1 & 0 & 0 & 0 & 0 & 0 \\
1 & 0 & 0 & 0 & 0 & 0 \\
1 & 0 & 0 & 0 & 0 & 0 \\
1 & 0 & 0 & 0 & 0 & 0 \\
1 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 1 \\
1 & 0 & 0 & 0 & 0 & 0 \\
1 & 0 & 0 & 0 & 0 & 0 \\
1 & 0 & 0 & 0 & 0 & 0 \\
1 & 0 & 0 & 0 & 0 & 0
\end{bmatrix}
\quad A
$$

Importance Sampling

- Say $s_7 = 1$ and $s_i = \frac{1}{10}$ for $i \neq 7$
- What is the probability that each row should be sampled?
- Proportional to importance!
- Say $p_7 = \frac{1}{2}$ and $p_i = \frac{1}{20}$ for $i \neq 7$

n

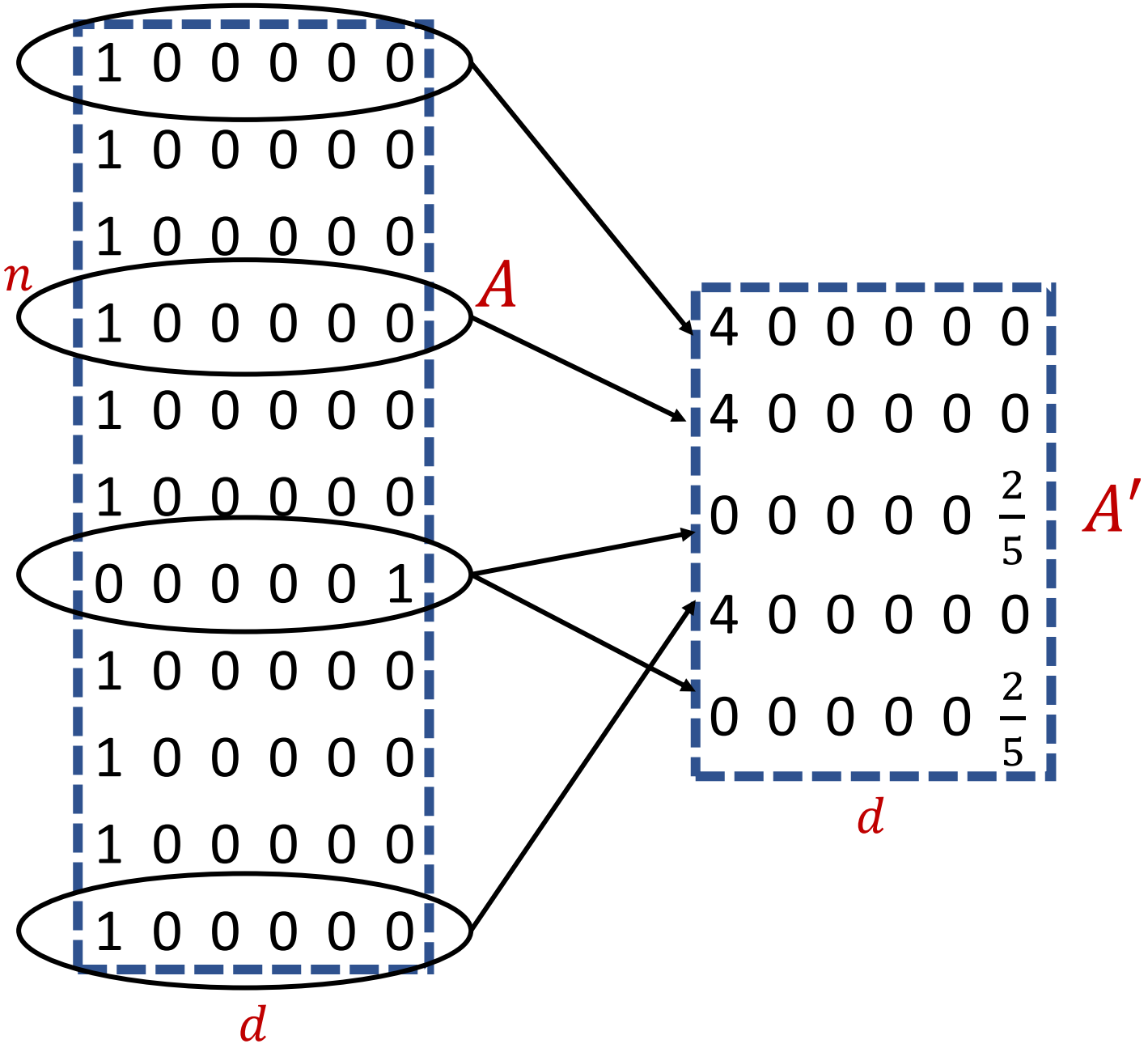
1	0	0	0	0	0
1	0	0	0	0	0
1	0	0	0	0	0
1	0	0	0	0	0
1	0	0	0	0	0
1	0	0	0	0	0
0	0	0	0	0	1
1	0	0	0	0	0
1	0	0	0	0	0
1	0	0	0	0	0
1	0	0	0	0	0

A

Importance Sampling

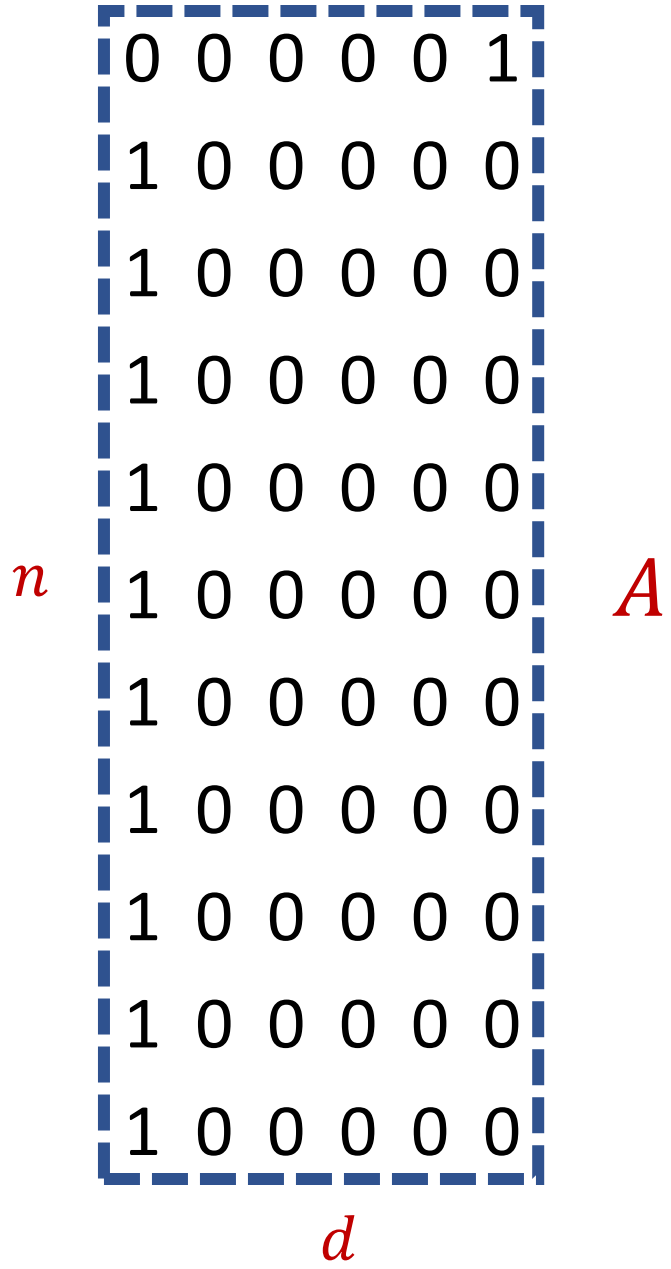
- Repeat m times:
  - Pick a row A_i of A with probability p_i and scale by $\frac{1}{mp_i}$

- Example with $n = 11$ and $m = 5$



Importance Sampling

- How big should m be?
- Have a measure s_i of "importance" for each row A_i
- Need $m = O(T \log n)$ rows, where $T = \sum s_i$



Sensitivity Sampling

- Terminology: s_i is called the sensitivity of the row A_i
- $T = \sum s_i$ is called the total sensitivity of A
- Need $m = O(T \log n)$ rows, where T can be much smaller than n

Sensitivity Sampling (Formal Theorem)

[Feldman, Schmidt, Sohler 2020] Let $C > 1$ be a universal constant and for each $i \in [n]$, let $q(x_i)$ be a C-approximation to the sensitivity $s(x_i)$ for any point x_i. Let $T = \sum_{i=1}^{n} q(x_i)$. Then sensitivity sampling

$$m = O\left(\frac{Tk}{\varepsilon^2}\log^2 k\right)$$

points with replacement, i.e., choosing each of the m points to be x_i with probability proportional to $q(x_i)$ and then rescaling by the sampling probability, outputs a $(1+\varepsilon)$-coreset for k-means clustering with probability $\frac{2}{3}$.

Sensitivity Sampling

- Pros: size is always better than uniform sampling and MUCH better on “worst-case” data

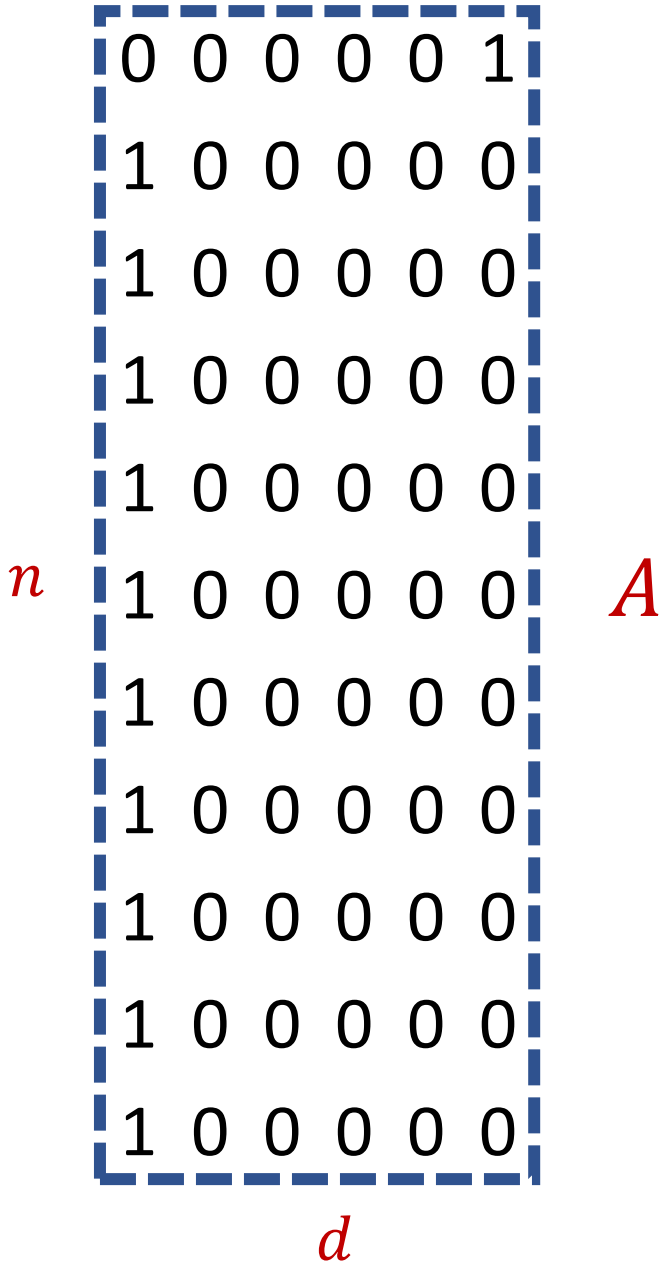
- Cons: estimating the sensitivities can be a difficult design choice, computing the total sensitivity can be mathematically challenging, larger runtime

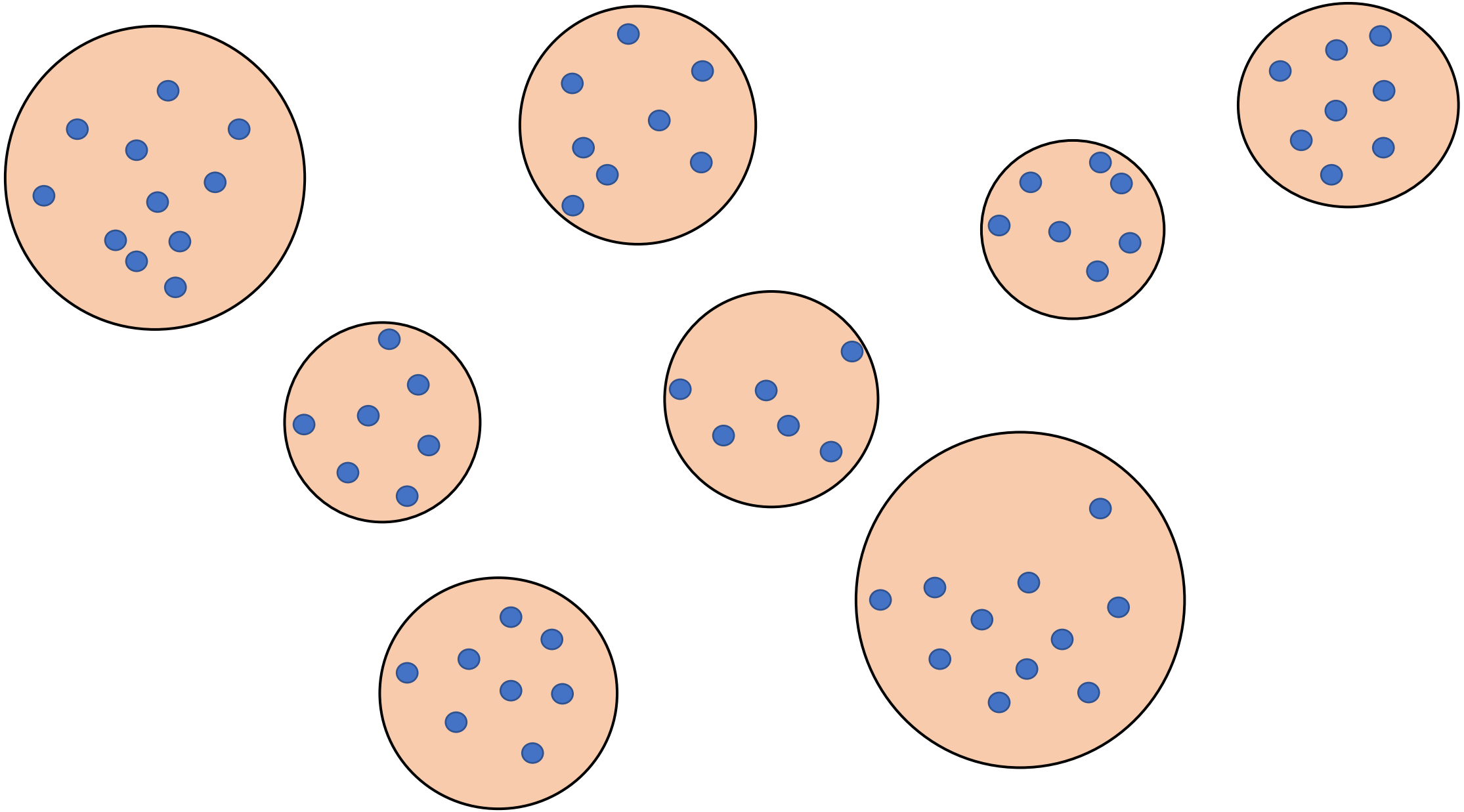
	Uniform Sampling	Sensitivity Sampling
Average-case size	X	X
Worst-case size		X
Ease of implementation Computation of probabilities	X	
Interpretability		X

Group / Stratified Sampling

- Separate the data into “groups” and then perform uniform sampling on each group

- If the data is uniform in each group, this is essentially sensitivity sampling





Coresets

- Where to apply them?
- Which sampling approach to use?
- Worst-case mathematical bounds

Coresets

- Where to apply them?

Published as a conference paper at ICLR 2020

DATA-INDEPENDENT NEURAL PRUNING VIA CORESETS

Ben Mussay
Computer Science Department
University of Haifa
Haifa, Israel
bengordoncshaifa@gmail.com

Margarita Osadchy
Computer Science Department
University of Haifa
Haifa, Israel
rita@cs.haifa.ac.il

Vladimir Braverman
Computer Science Department
Johns Hopkins University
Baltimore, MD., USA
vova@cs.jhu.edu

Samson Zhou
Computer Science Department
Carnegie Mellon University
Pittsburgh, IN., USA
samsonzhou@gmail.com

Dan Feldman
Computer Science Department
University of Haifa
Haifa, Israel
dannyf.post@gmail.com

Coresets

- Which sampling approach to use?

APPROXIMATING ANY FUNCTION VIA CORESET FOR RADIAL BASIS FUNCTIONS: TOWARDS PROVABLE DATA SUBSET SELECTION FOR EFFICIENT NEURAL NETWORKS TRAINING

ABSTRACT

Radial basis function neural networks (*RBFNN*) are well-known for their capability to approximate any continuous function on a closed bounded set with arbitrary precision given enough hidden neurons. Coreset is a small weighted subset of an input set of items, that provably approximates their loss function for a given set of queries (models, classifiers, etc.). In this paper, we suggest the first coreset construction algorithm for *RBFNNs*, i.e., a small weighted subset which approximates the loss of the input data on any radial basis function network and thus approximates any function defined by an *RBFNN* on the big input data. This is done by constructing coresets for radial basis and Laplacian loss functions. We use our coreset to suggest a provable data subset selection algorithm for training deep neural networks, since our coreset approximates every function, it should approximate the gradient of each weight in a neural network as it is defined as a function on the input. Experimental results on function approximation and dataset subset selection on popular network architectures and data sets are presented, demonstrating the efficacy and accuracy of our coreset construction.

Coresets

- Worst-case mathematical bounds

New Coresets for Projective Clustering and Applications

Murad Tukan * Xuan Wu† Samson Zhou‡

Vladimir Braverman§ Dan Feldman▽

March 10, 2022

Abstract

(j, k)-projective clustering is the natural generalization of the family of k-clustering and j-subspace clustering problems. Given a set of points P in $\mathbb{R}^d$, the goal is to find k flats of dimension j, i.e., affine subspaces, that best fit P under a given distance measure. In this paper, we propose the first algorithm that returns an L_∞ coreset of size polynomial in d. Moreover, we give the first strong coreset construction for general M-estimator regression. Specifically, we show that our construction provides efficient coreset constructions for Cauchy, Welsch, Huber, Geman-McClure, Tukey, $L_1 - L_2$, and Fair regression, as well as general concave and power-bounded loss functions. Finally, we provide experimental results based on real-world datasets, showing the efficacy of our approach.