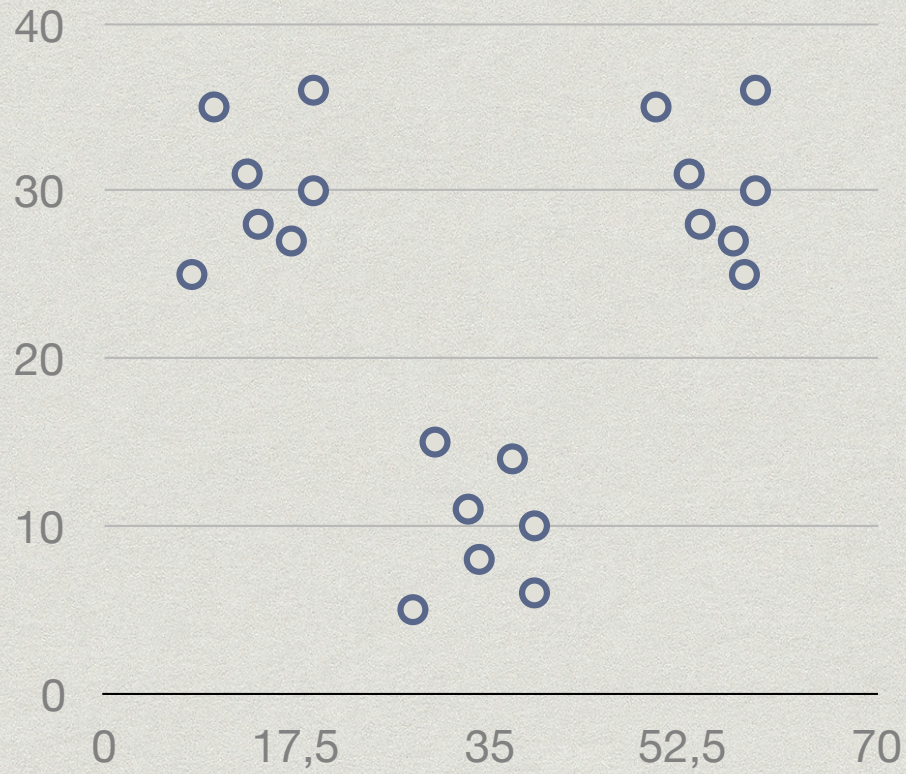
CLUSTERING

ALESSANDRO PANCONESI, SAPIENZA

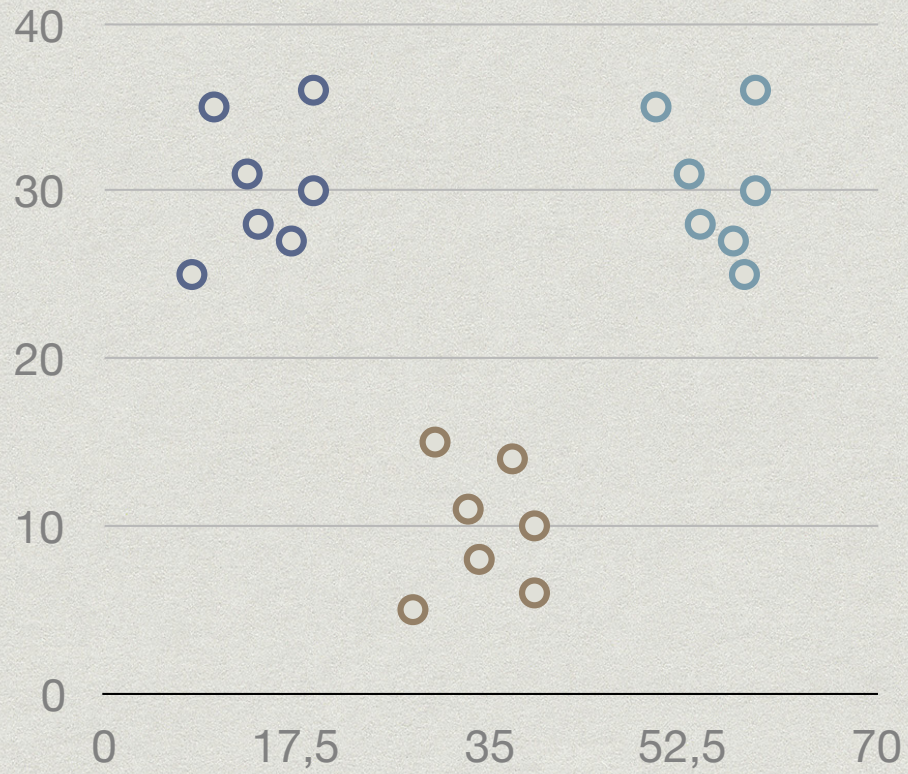
What is Clustering?

- A grouping of data objects such that, objects in the same group are similar (or close) while objects in different groups are dissimilar (or far apart)



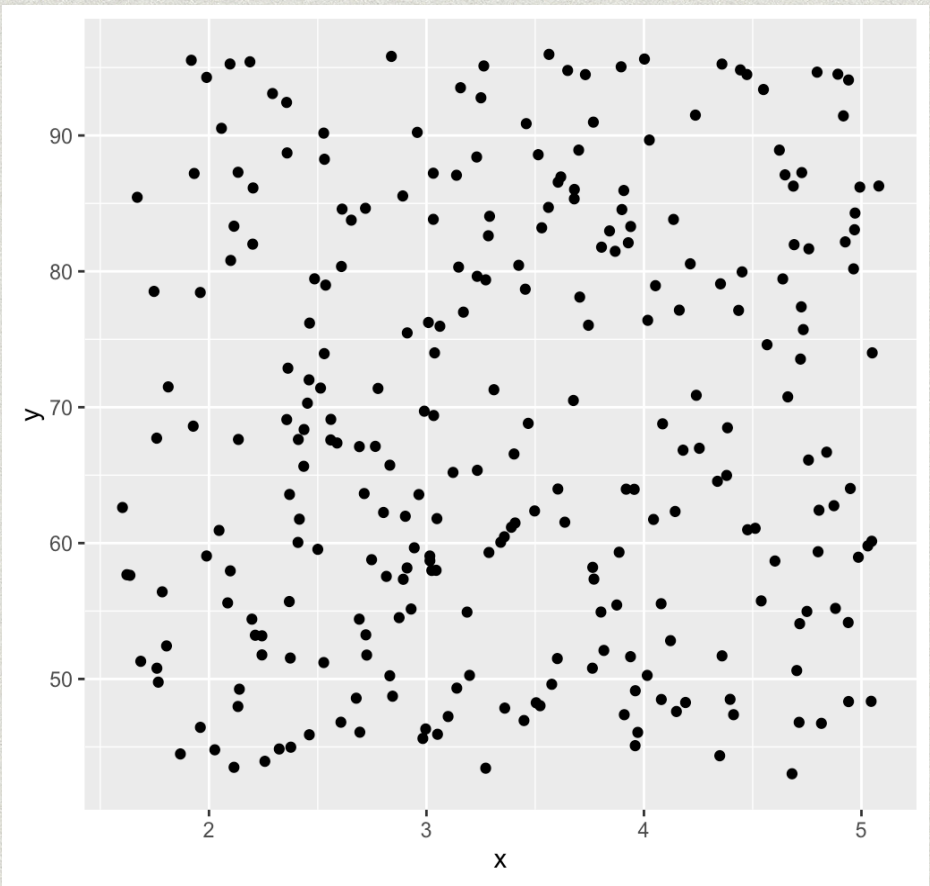
What is Clustering?

- A grouping of data objects such that, objects in the same group are similar (or close) while objects in different groups are dissimilar (or far apart)

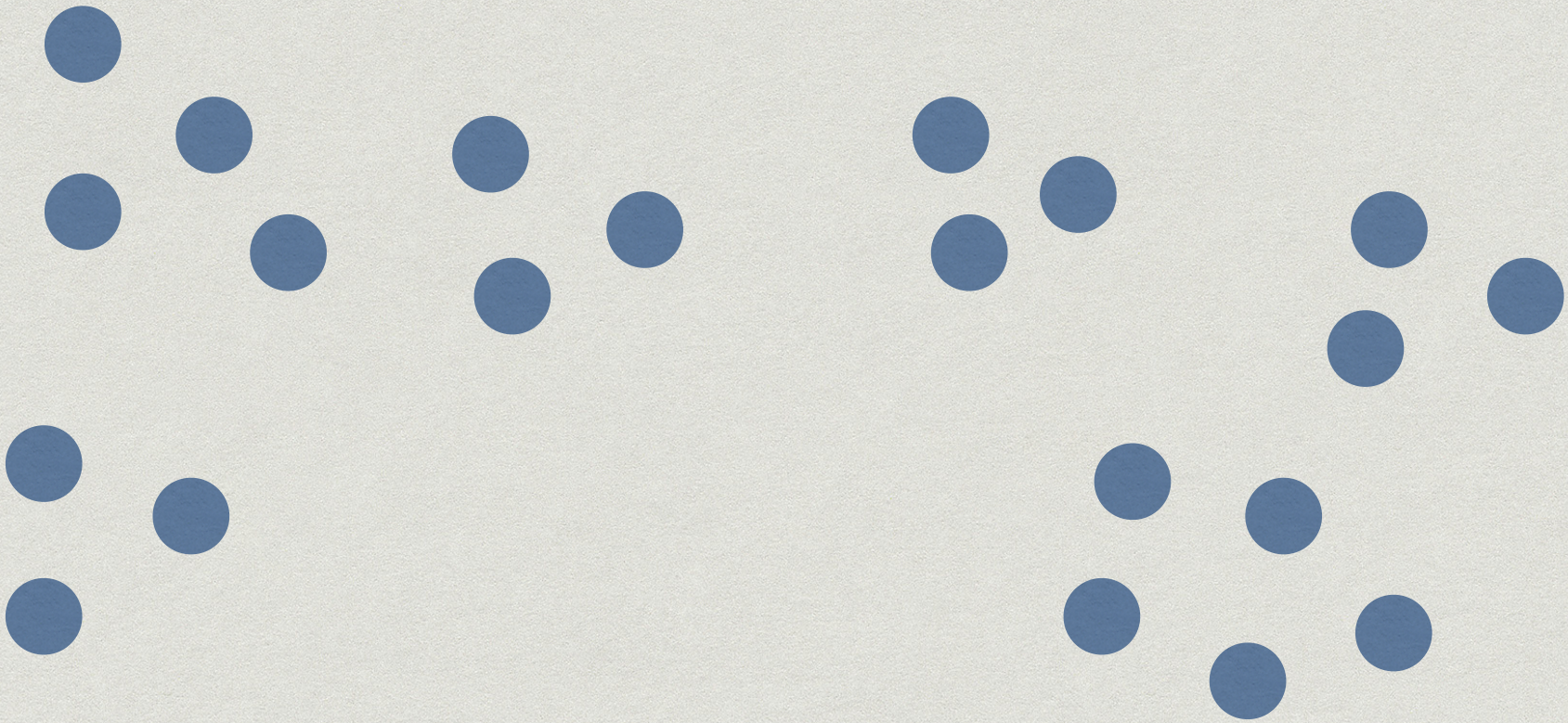


What is Clustering?

- Clustering is an intuitively appealing idea, but it is far from being well-defined, or for that matter, well-definable

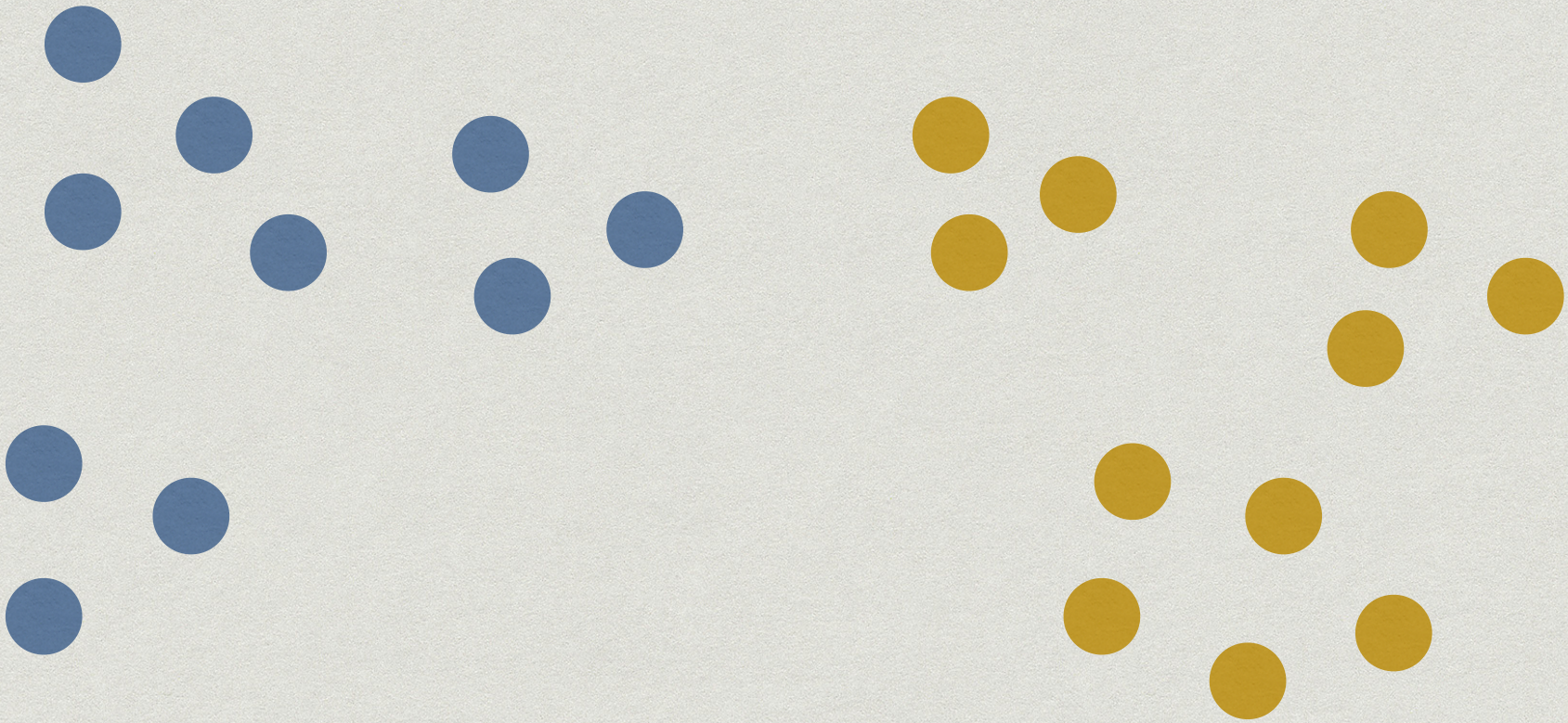


What is Clustering?



What is Clustering?

Two clusters?



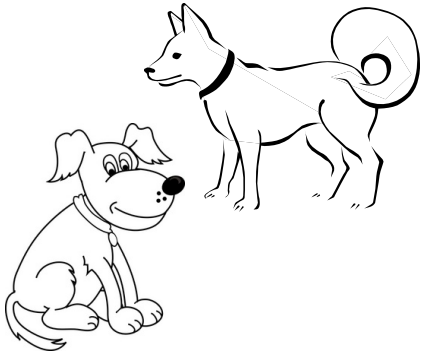
What is Clustering?

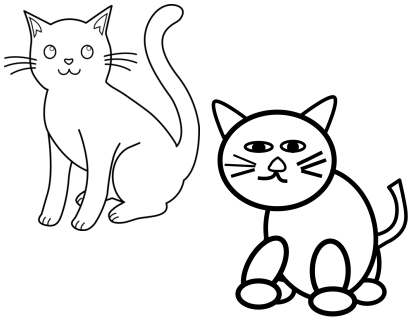
Four clusters?

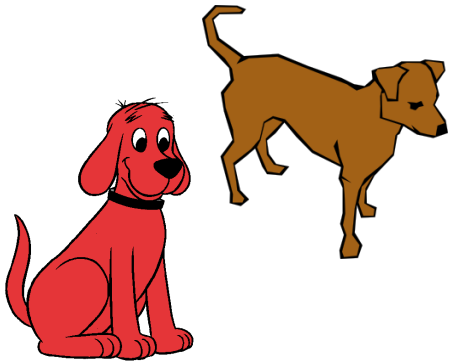
What is Clustering?

Six clusters?

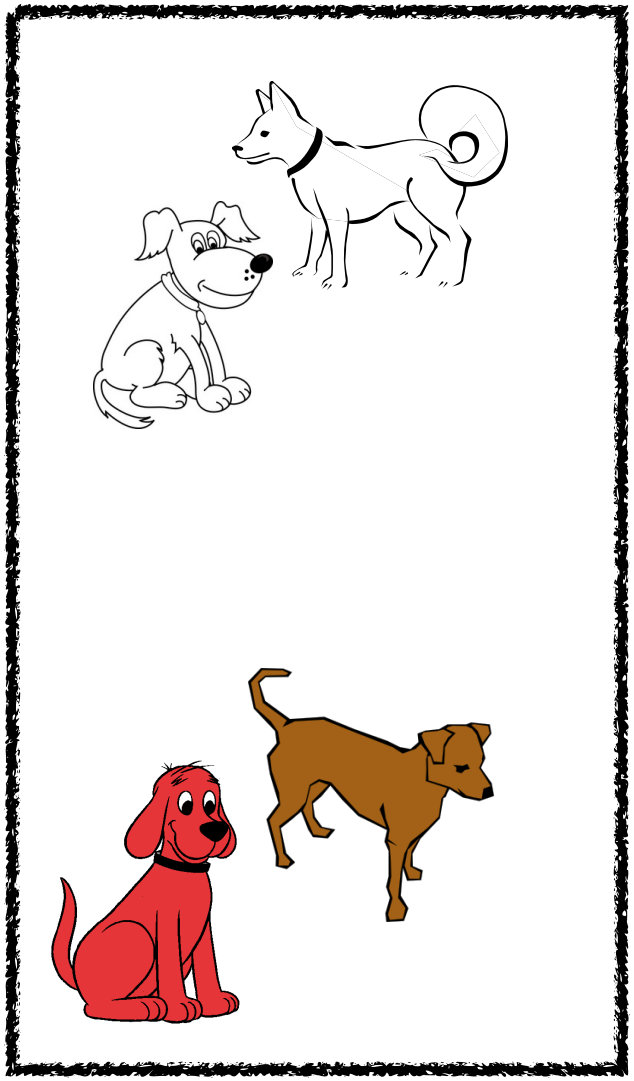
What's similar?

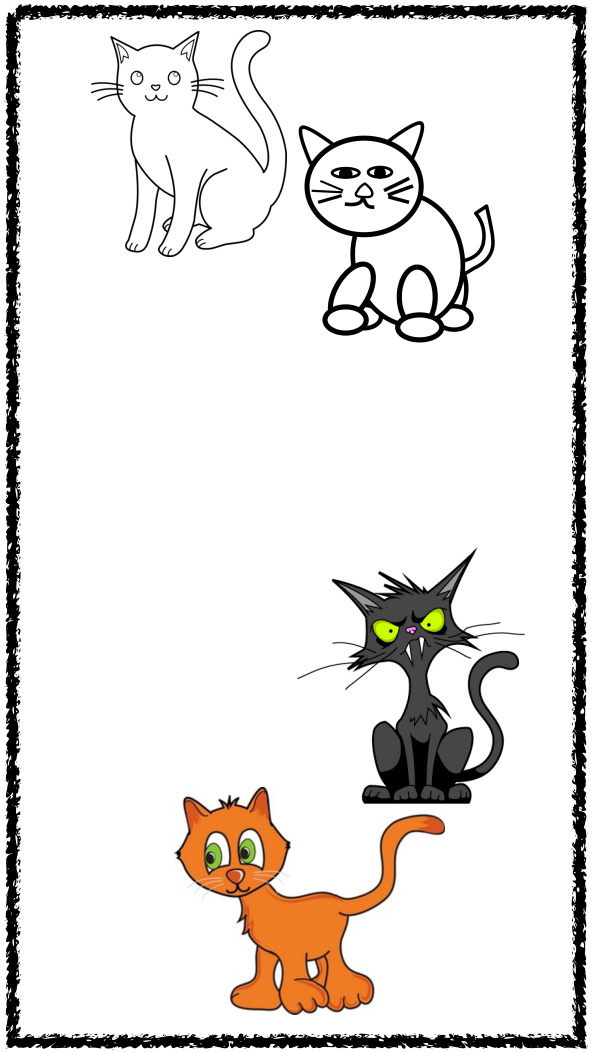






What's similar?





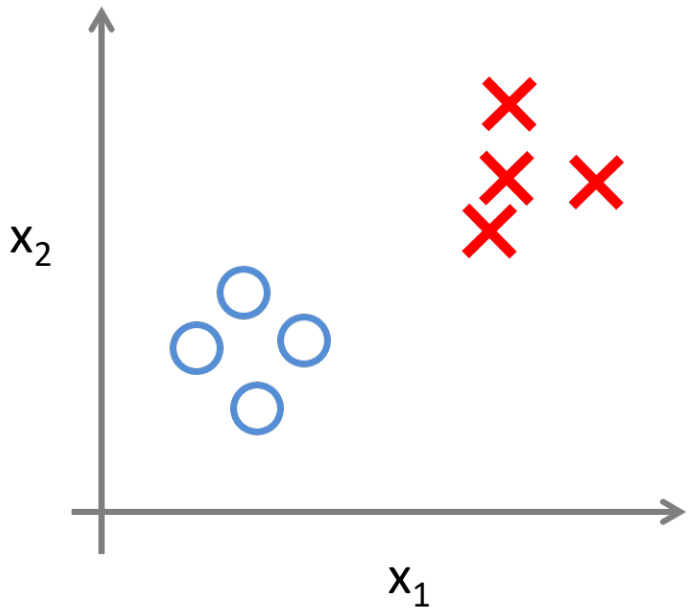
What's similar?

What is clustering?

- Still, clustering is extremely useful
- …with a huge array of applications in all sorts of data analysis

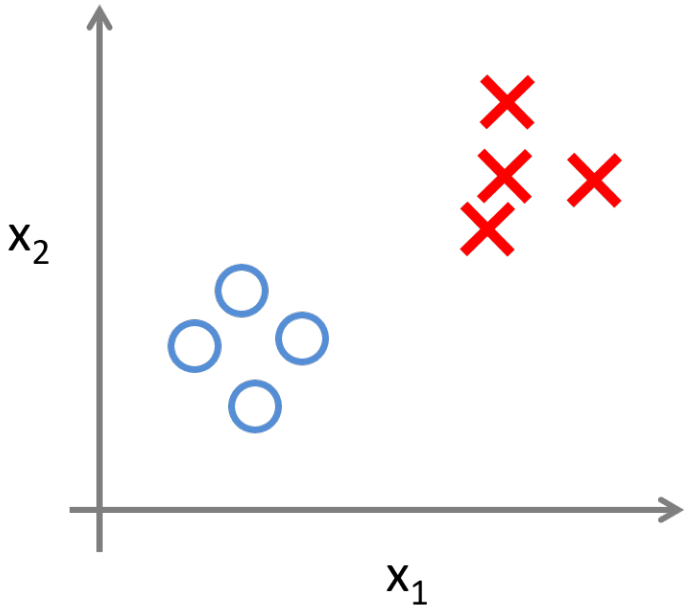
Supervised vs. Unsupervised

Supervised Learning

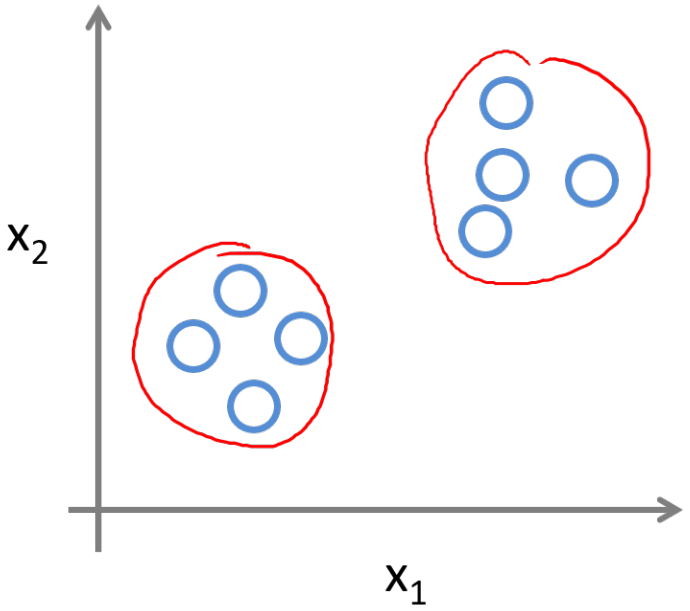


Supervised vs. Unsupervised

Supervised Learning



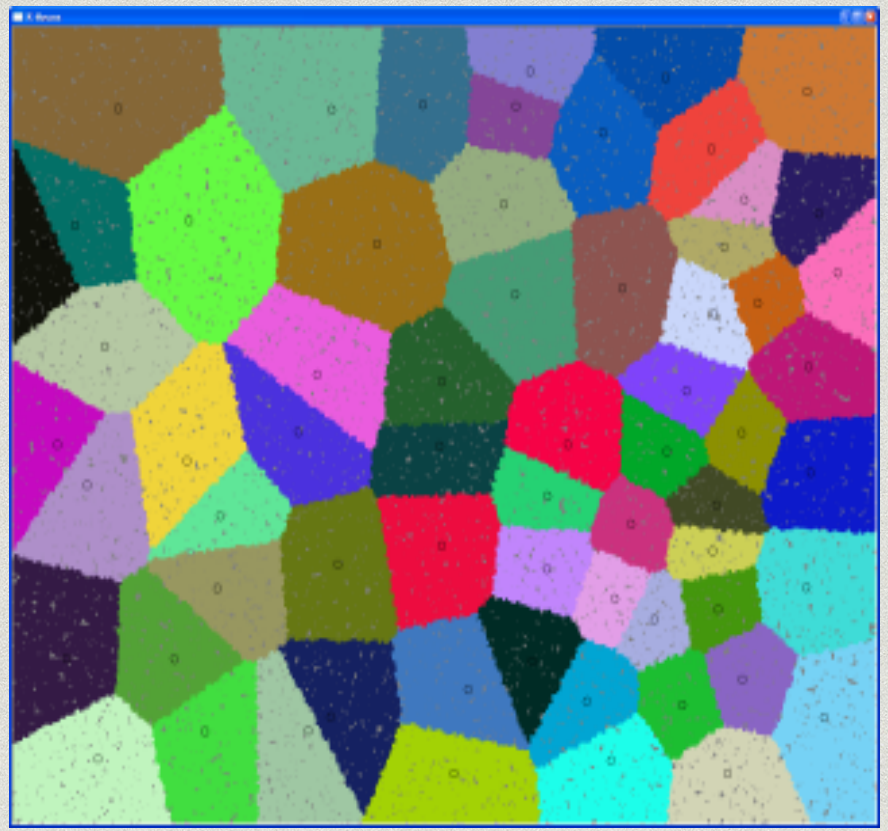
Unsupervised Learning



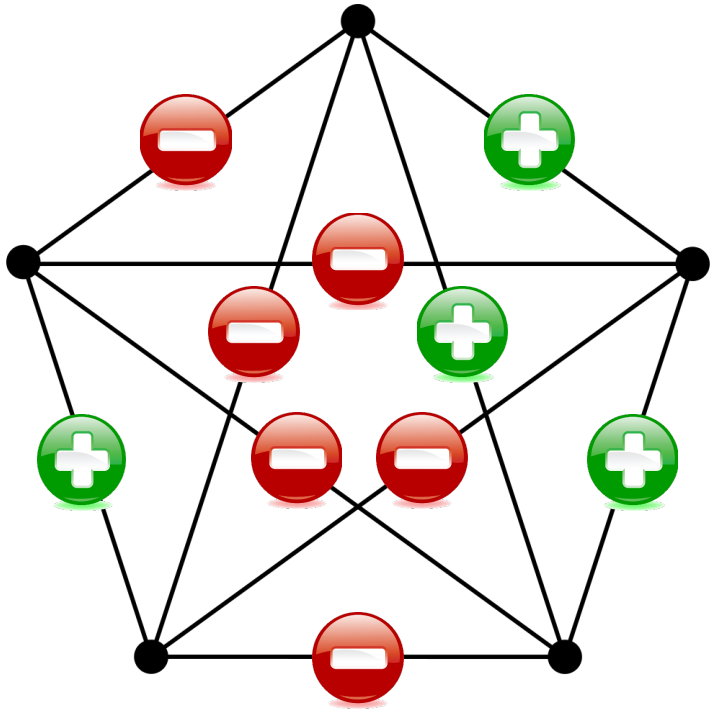
THE CLUSTERING ZOO

The Clustering Zoo

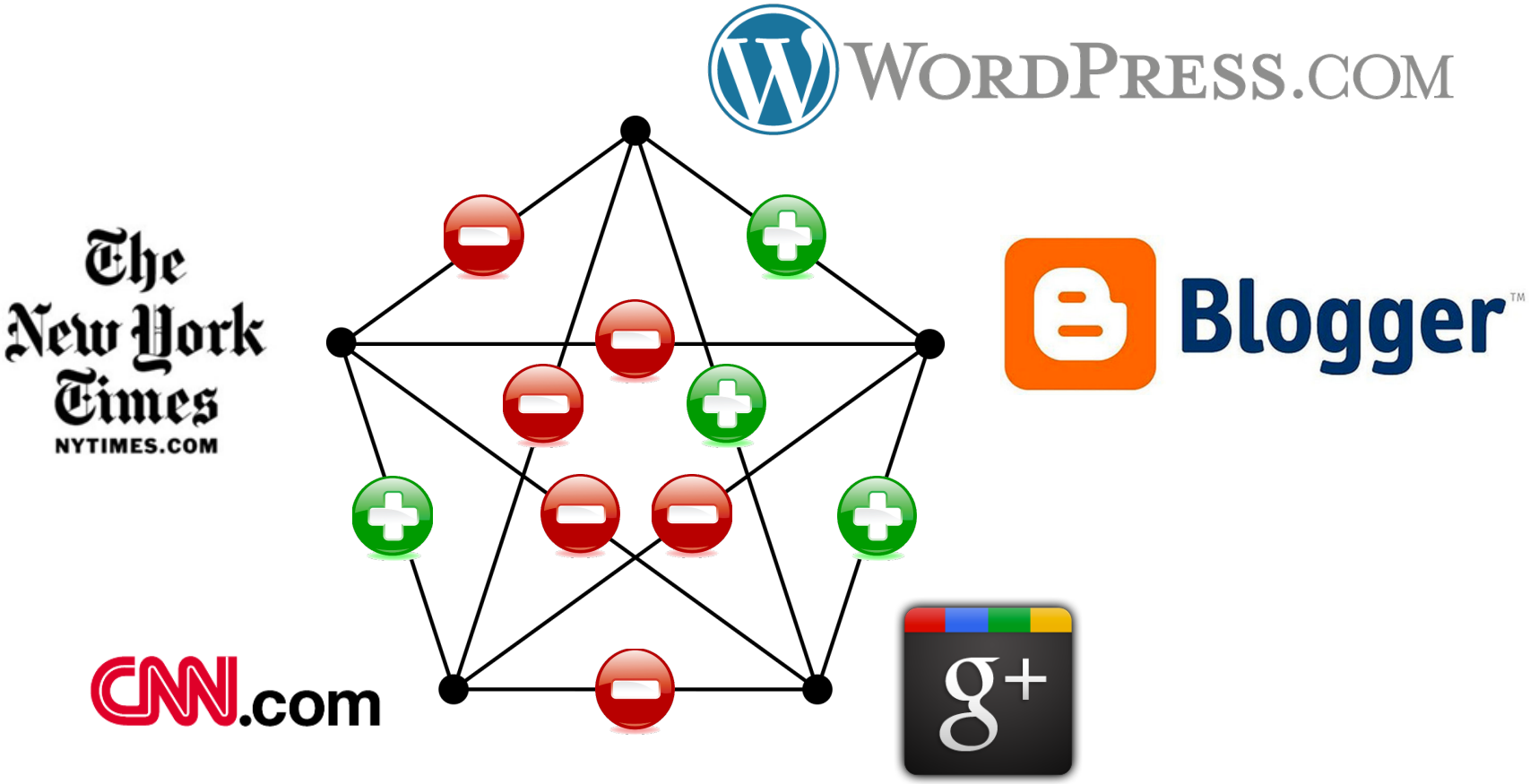
- There are dozens of different notions of “clustering”
- Here, we will consider only clusterings that partition the input data (a graph or points in space)



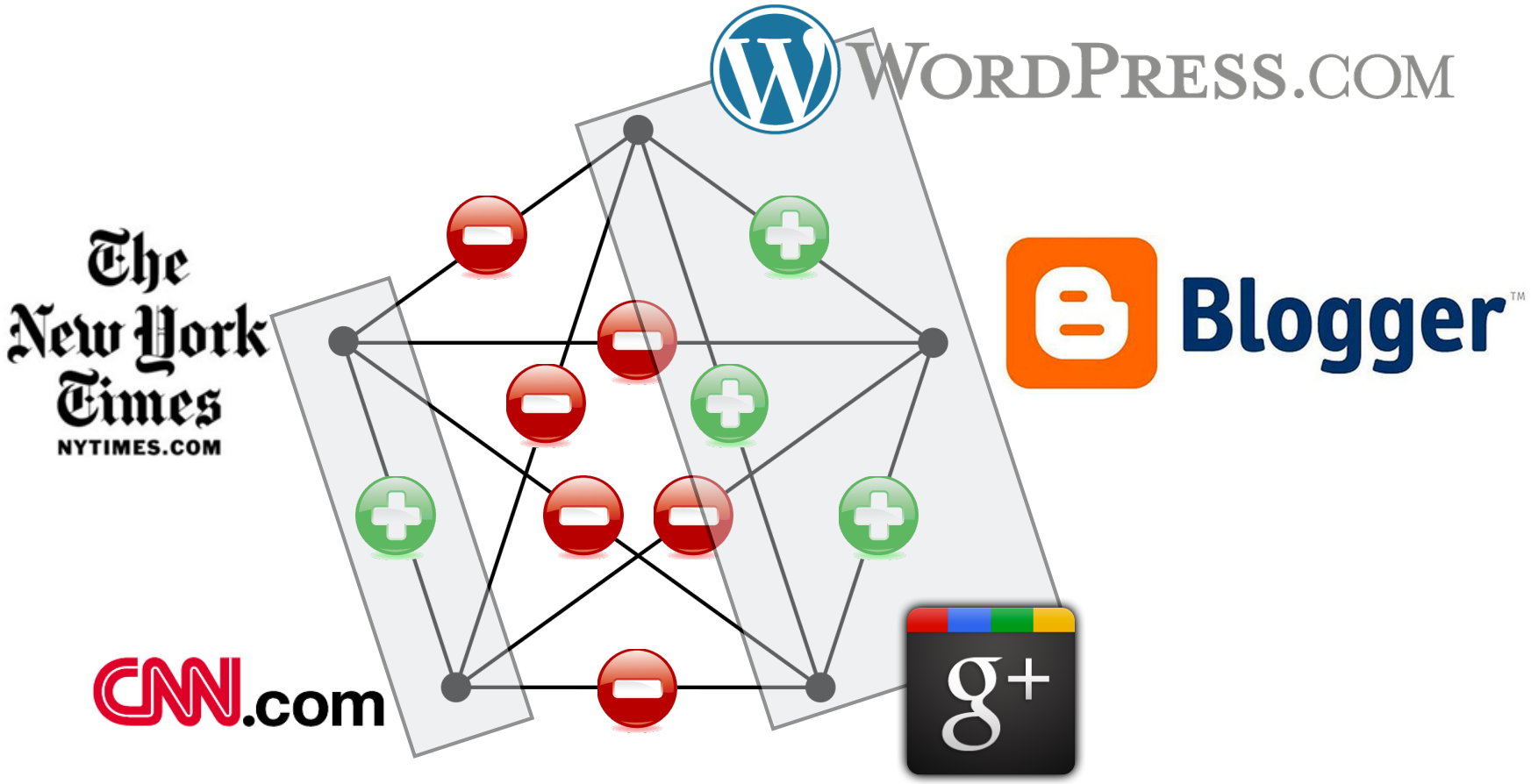
Correlation Clustering



Correlation Clustering



Correlation Clustering



Correlation Clustering

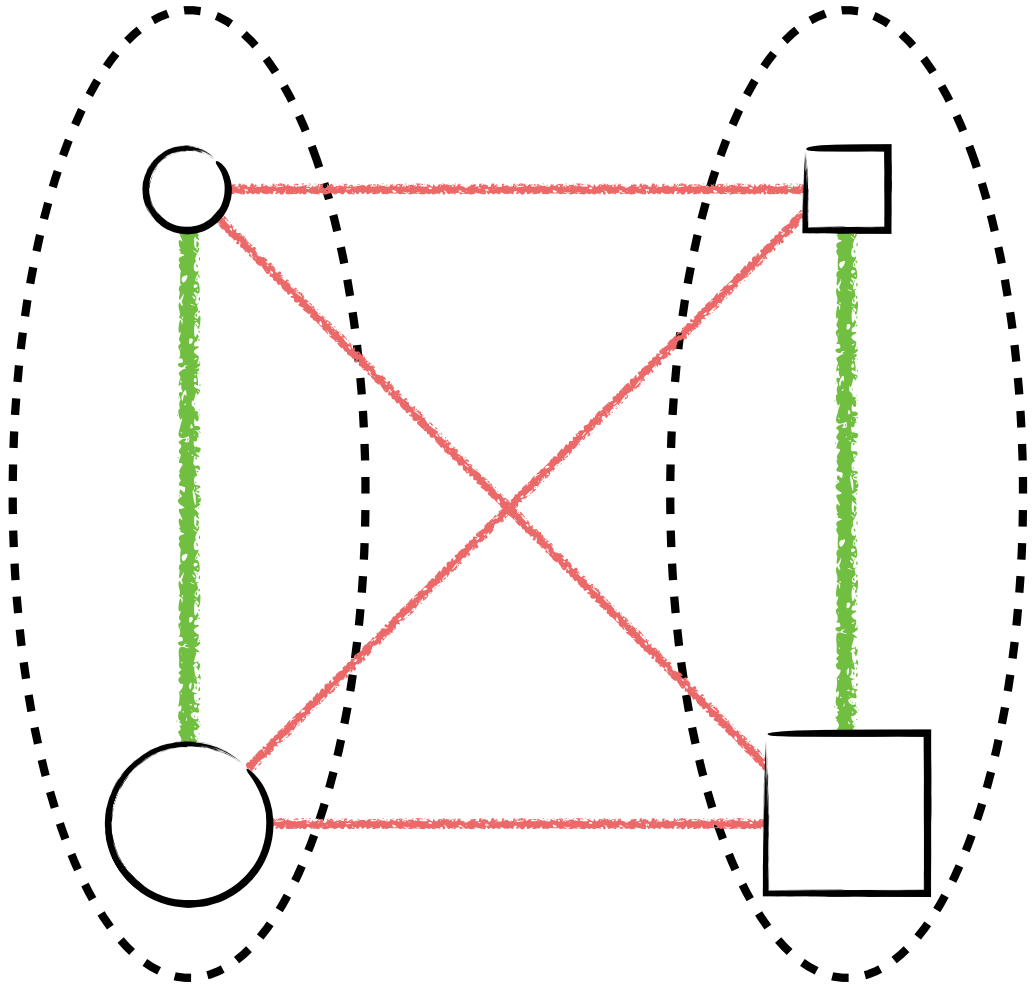
The goal is to partition the elements in
disjoint clusters
so as to **minimize** the number of **mistakes**

Correlation Clustering

The goal is to partition the elements in
disjoint clusters
so as to **minimize** the number of **mistakes**

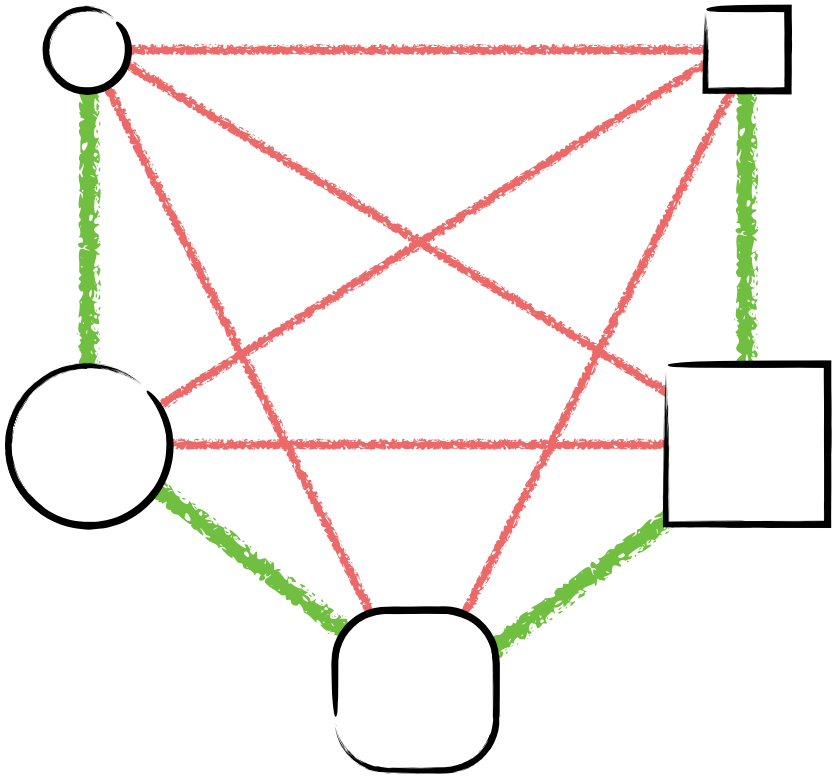
The **number of clusters**
is not decided a priori

Correlation Clustering



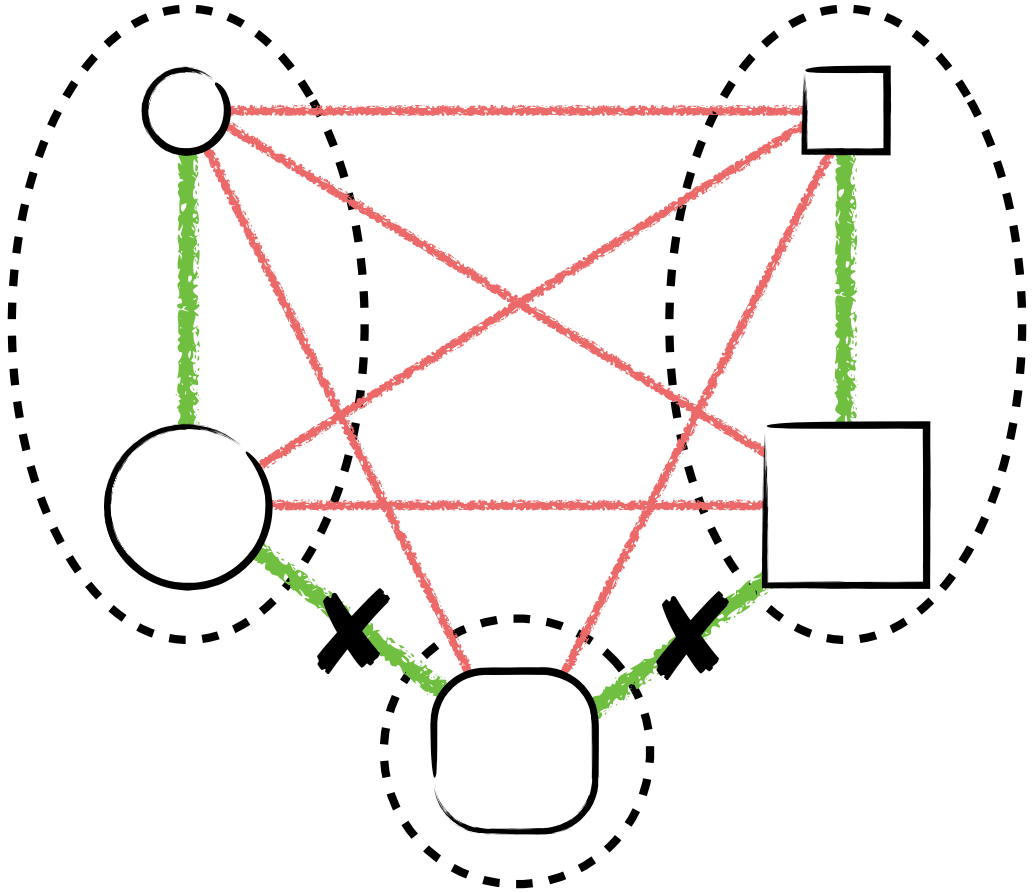
"Perfect clustering"
(0 mistakes)

Correlation Clustering



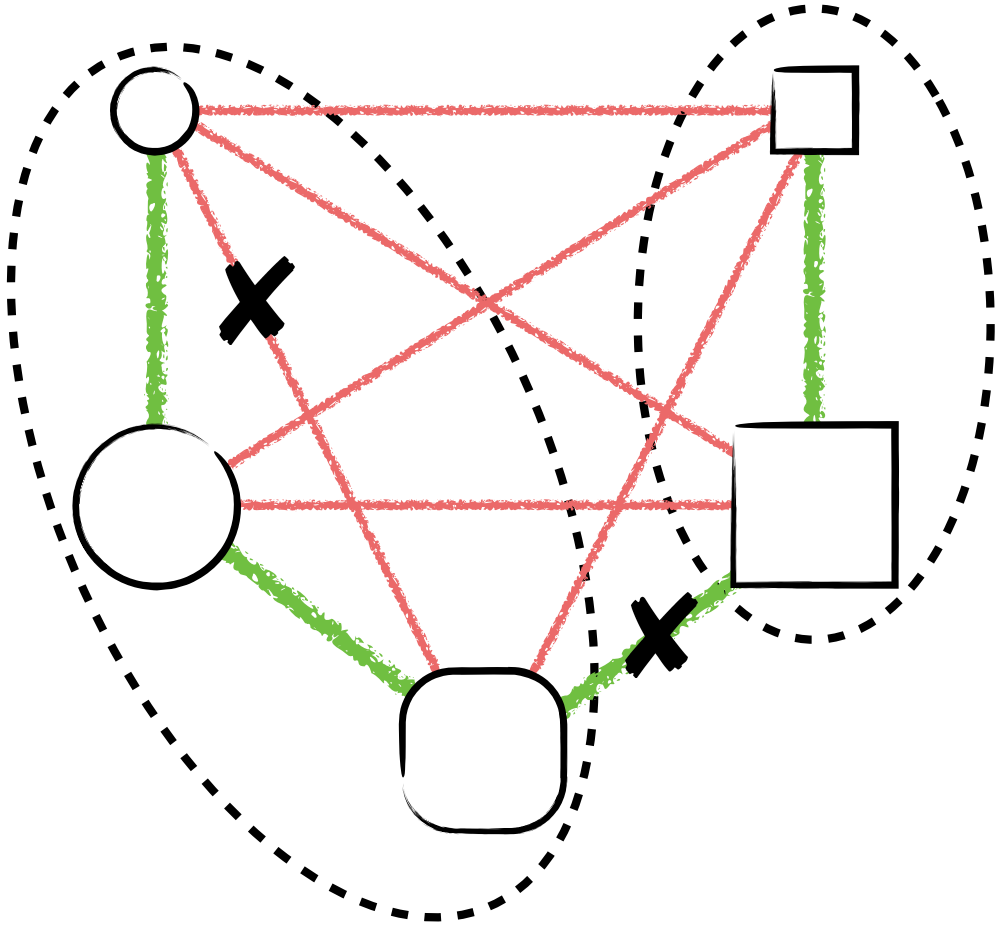
In this instance you cannot have zero mistakes

Correlation Clustering



In this instance you cannot have zero mistakes

Correlation Clustering



In this instance you cannot have zero mistakes

k-means Clustering

- We are given a set S of n points in R^d
- Assume k is also given
- Find k points $c_1,..,c_k$, called centers (or means) such that the following cost is minimised

$$\text{cost}(c_1, \ldots, c_k) := \sum_{i=1}^{n} \min_j d(x_i, c_j)^2$$

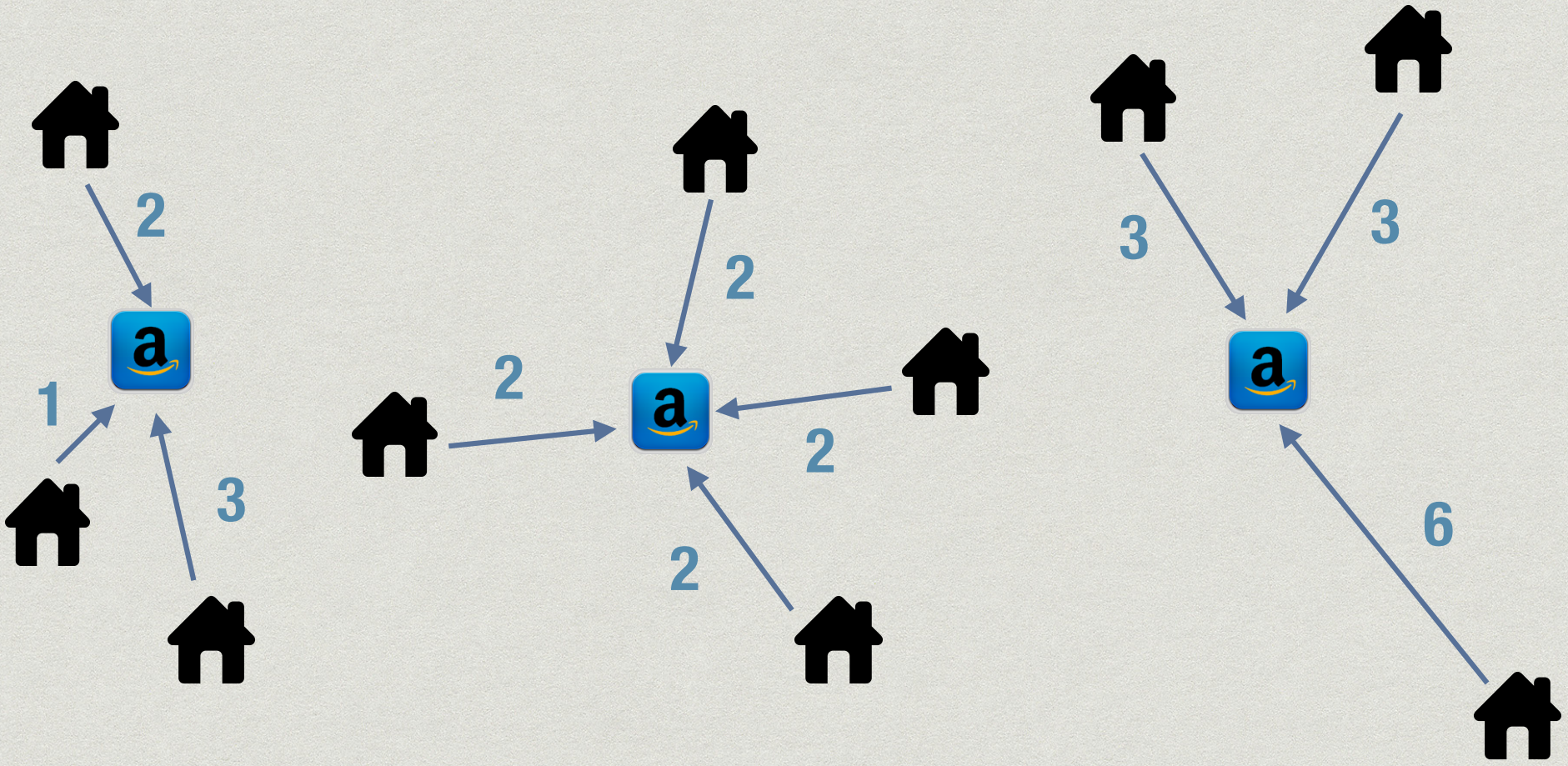
k-means Clustering

$$\text{cost}(c_1, \ldots, c_k) := \sum_{i=1}^{n} \min_j d(x_i, c_j)^2$$

k-means Clustering

$$\text{cost}(c_1, \ldots, c_k) := \sum_{i=1}^{n} \min_j d(x_i, c_j)^2$$

k-means Clustering



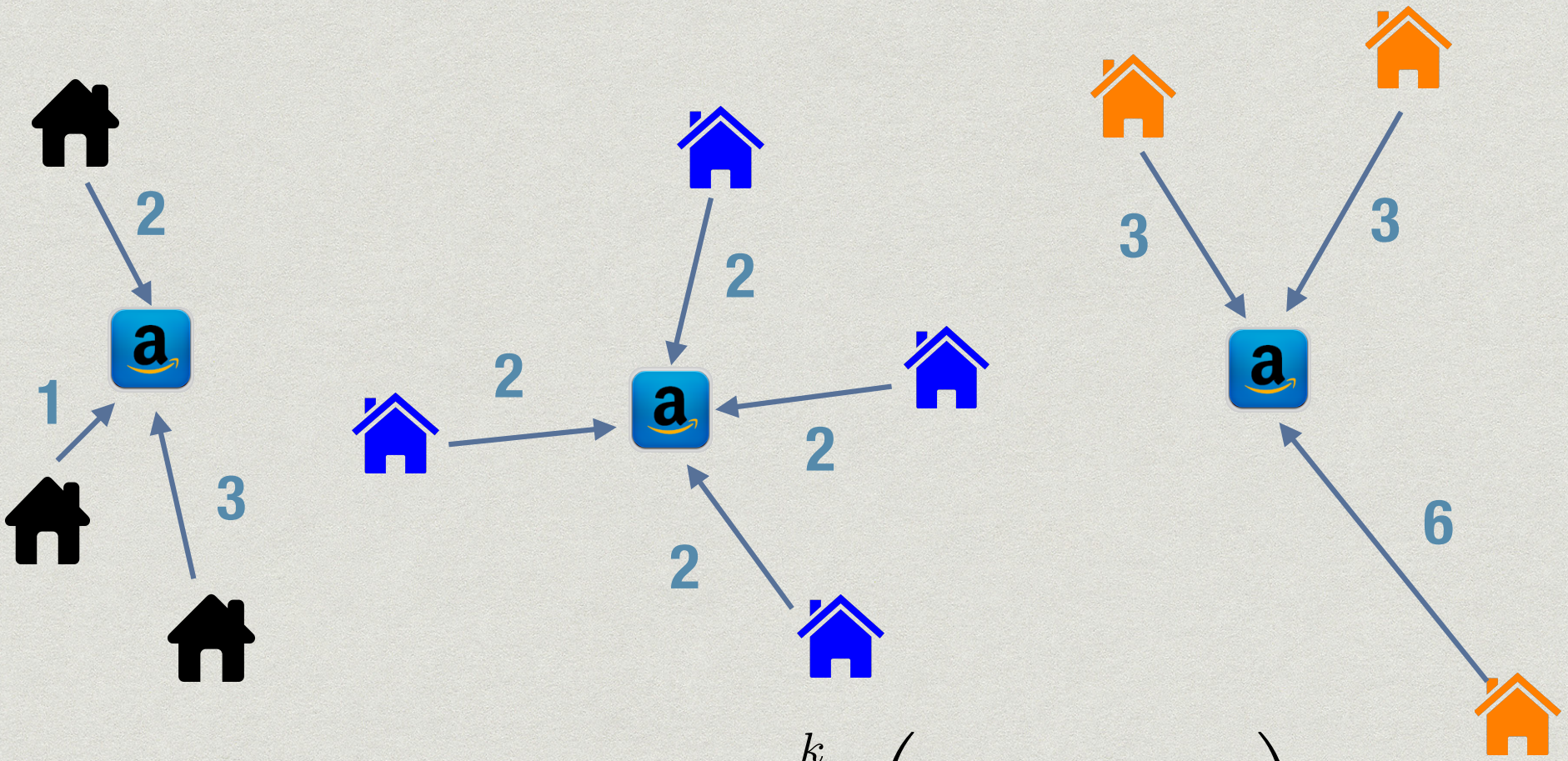
$$cost(c_1, c_2, c_3) = 84$$

k-means Clustering

- Equivalently, given a set S of n points in R^d and an integer k…
- find $c_1,..,c_k$ and partition S into $S_1,..,S_k$ by assigning each x in S_i to c_i, in such a way the following cost is minimised

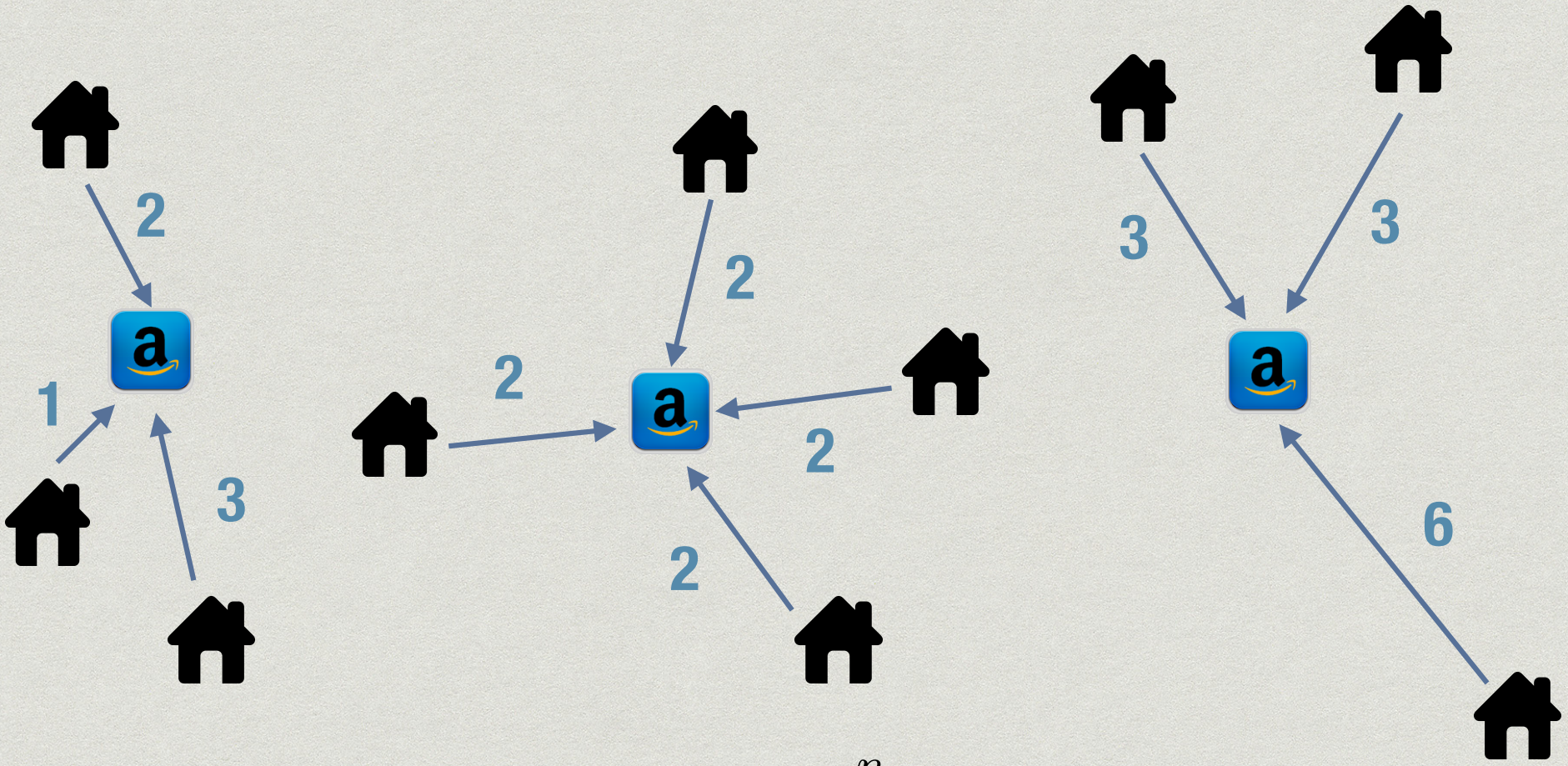
$$\text{cost}(c_1, \ldots, c_k) := \sum_{i=1}^{k} \left(\sum_{x \in S_i} d(x, c_i)^2 \right)$$

k-means Clustering



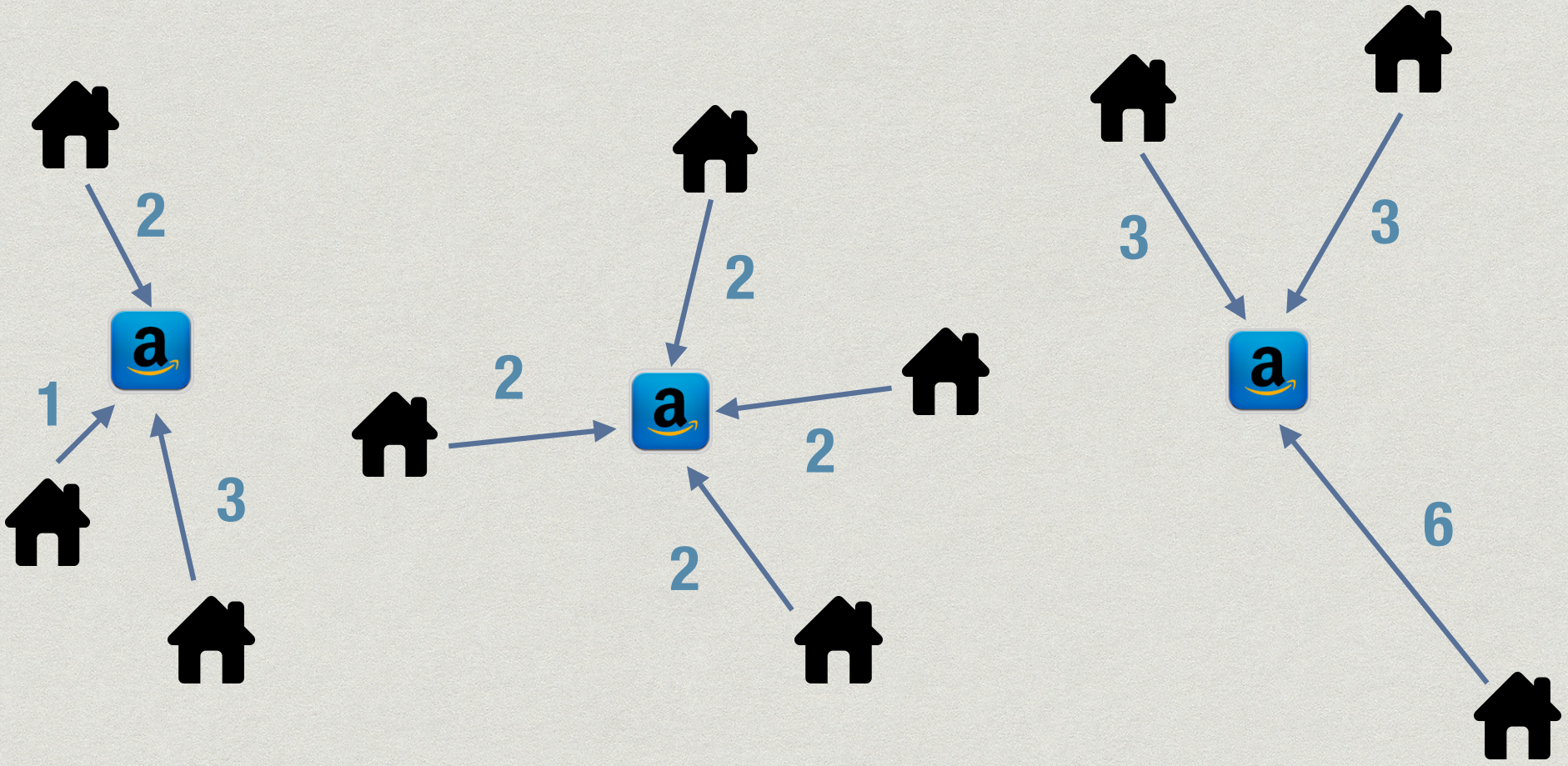
$$\text{cost}(c_1, \ldots, c_k) := \sum_{i=1}^{k} \left(\sum_{x \in S_i} d(x, c_i)^2 \right)$$

k-median Clustering



$$\text{cost}(c_1, \ldots, c_k) := \sum_{i=1}^{n} \min_j d(x_i, c_j)$$

k-median Clustering



$$cost(c_1, c_2, c_3) = 26$$

k-median Clustering

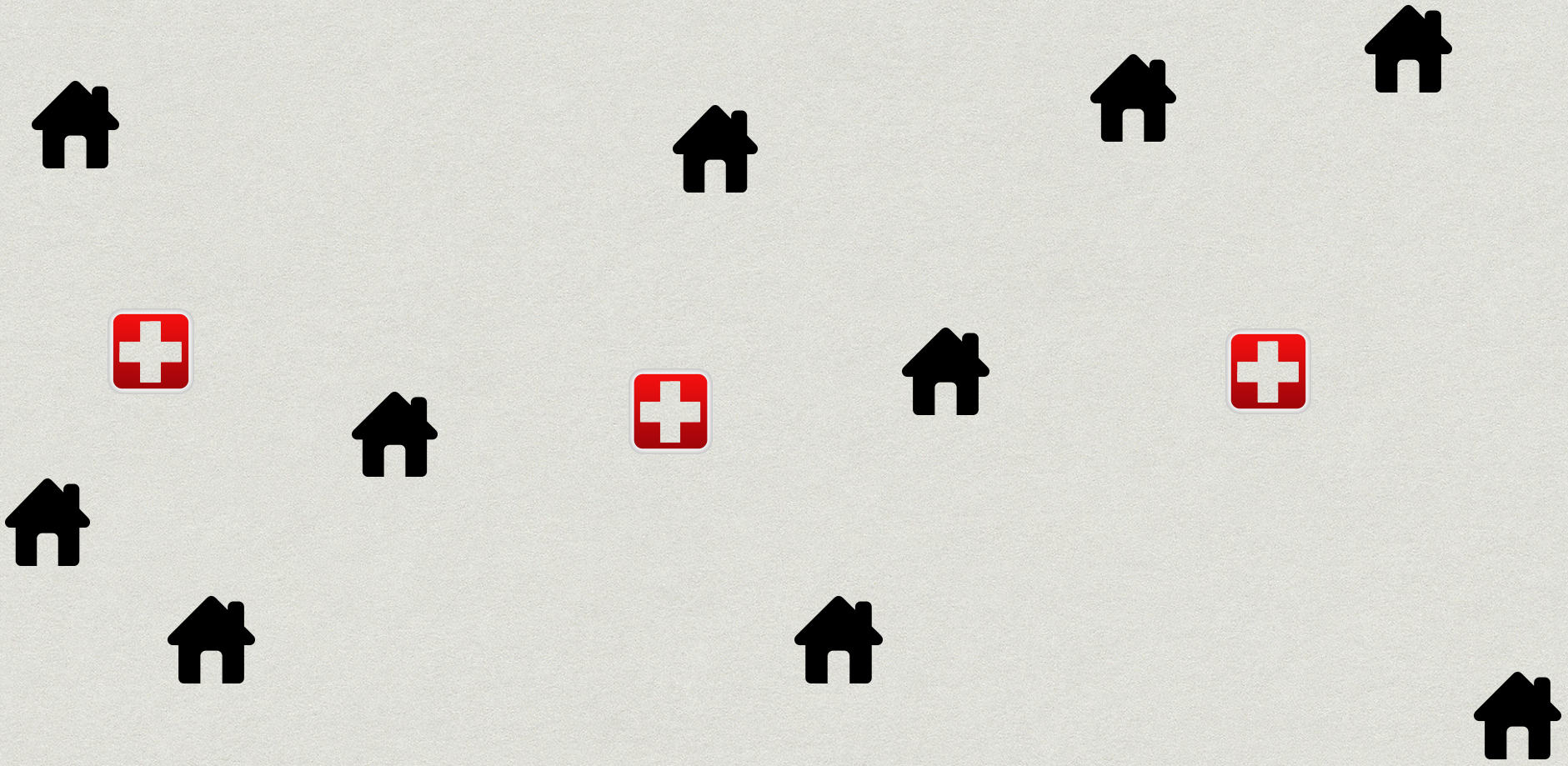
- Same input as in k-means clustering, but objective function becomes

$$\text{cost}(c_1, \ldots, c_k) := \sum_{i=1}^{n} \min_j d(x_i, c_j)$$

k-center Clustering

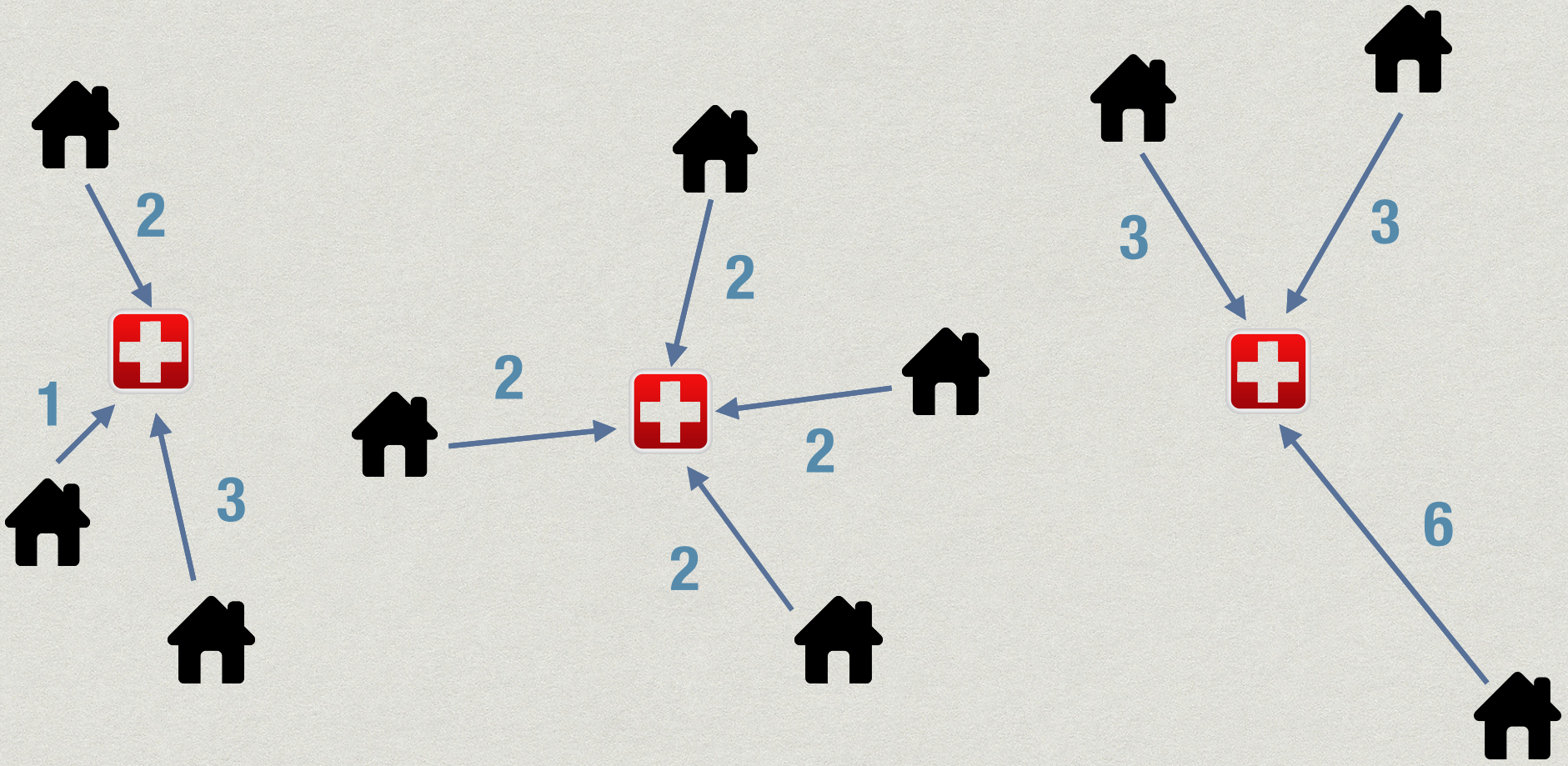
$$\text{cost}(c_1, \ldots, c_k) := \max_{x \in S} \min_{j} d(x, c_i)$$

k-center Clustering



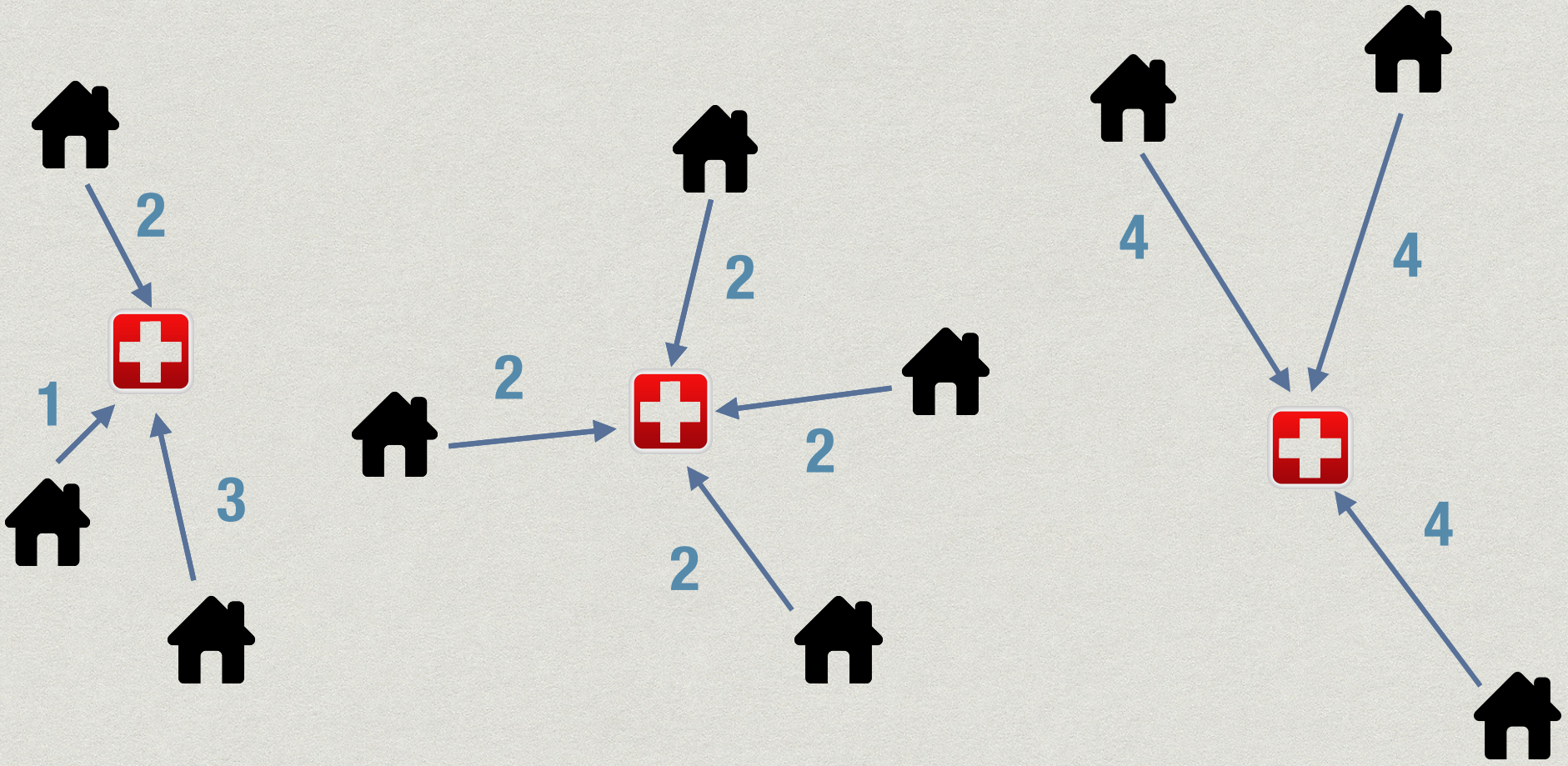
$$\text{cost}(c_1, \ldots, c_k) := \max_{x \in S} \min_{j} d(x, c_i)$$

k-center Clustering



$$\text{cost}(c_1, \ldots, c_k) := \max_{x \in S} \min_{j} d(x, c_i)$$

k-center Clustering



$$\text{cost}(c_1, \ldots, c_k) := \max_{x \in S} \min_{j} d(x, c_i)$$

k-center Clustering

- We want to build k facilities (say hospitals) in such a way that no point in S is too penalised, i.e. we want to minimise the maximum distance between every point and its closest center

$$\text{cost}(c_1, \ldots, c_k) := \max_{x \in S} \min_{j} d(x, c_i)$$

Algorithms for clustering

K-Means / Lloyds' Algorithm

- We are given *n* vectors on *d* dimensions;
  we aim to select *k* clusters

- Choose *k* vectors arbitrarily (eg uniformly at random) as *centroids*

K-Means / Lloyds' Algorithm

- We are given *n* vectors on *d* dimensions;
  we aim to select *k* clusters
- Choose *k* vectors arbitrarily (eg uniformly at random) as *centroids*
- Until *stability:*
  - create an empty cluster for each centroid;
  - add each vector to the cluster of the closest centroid;
  - for each cluster, compute its new centroid (the mean of its vectors).

K-Means / Lloyds' Algorithm

- We are given *n* vectors on *d* dimensions;
  we aim to select *k* clusters
- Choose *k* vectors arbitrarily (eg uniformly at random) as *centroids*
- Until *stability:*
  - create an empty cluster for each centroid;
  - add each vector to the cluster of the closest centroid;

$$\mathrm{dist}(x, y) = \sum (x(i) - y(i))^2$$

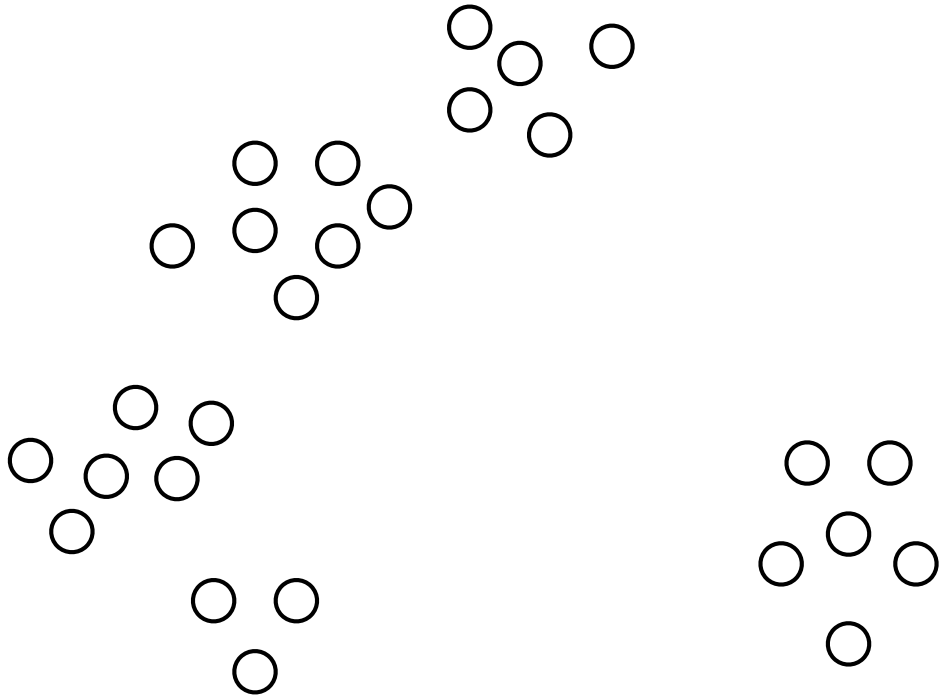
  - for each cluster, compute its new centroid (the mean of its vectors).

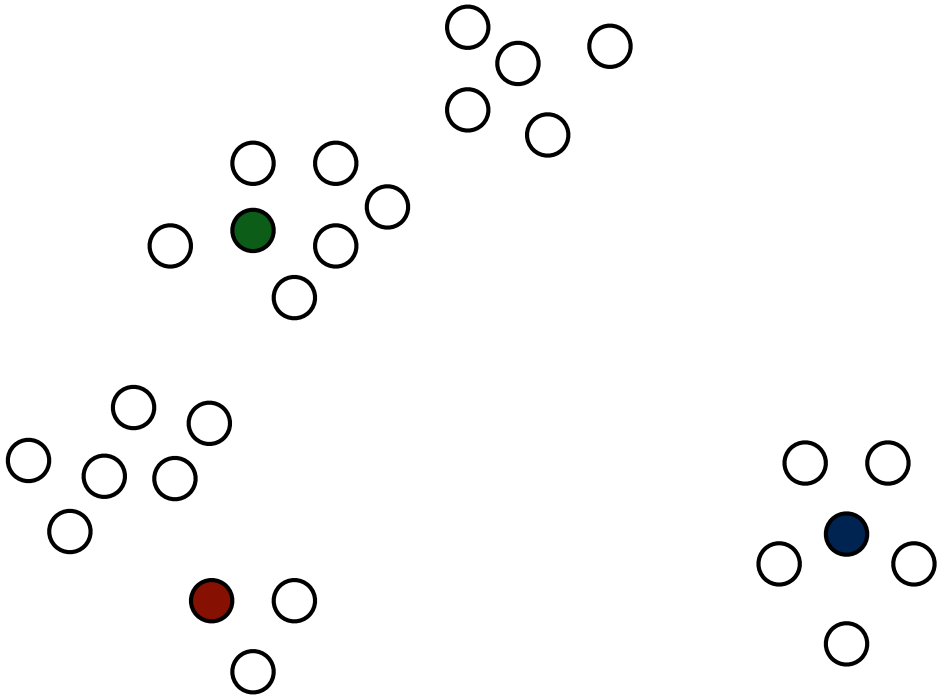
K-Means / Lloyds' Algorithm

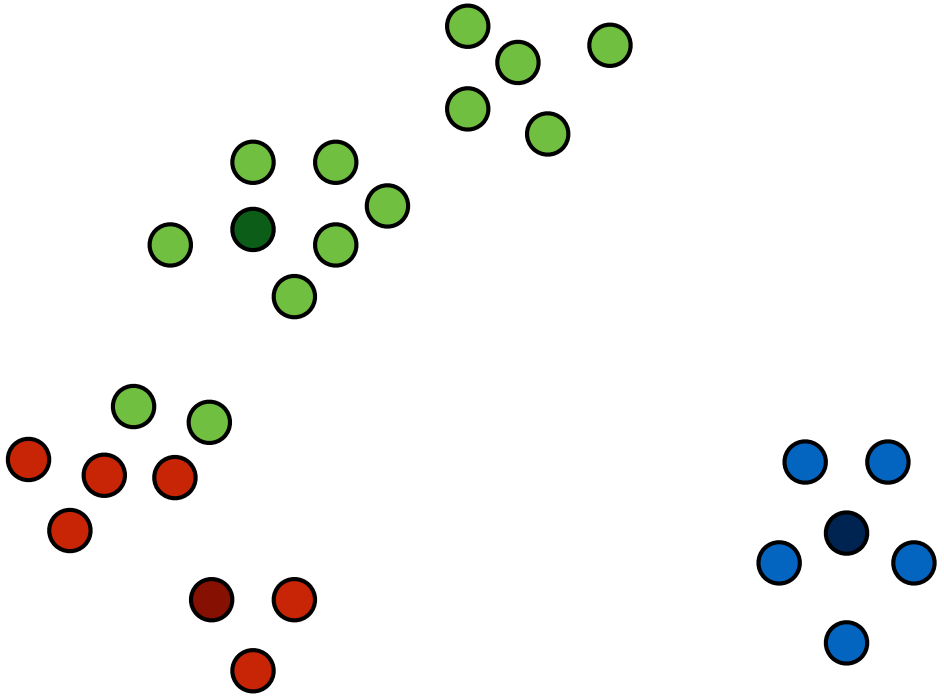
- We are given *n* vectors on *d* dimensions;
  we aim to select *k* clusters
- Choose *k* vectors arbitrarily (eg uniformly at random) as *centroids*
- Until *stability:*
  - create an empty cluster for each centroid;
  - add each vector to the cluster of the closest centroid; $O(nkd)$

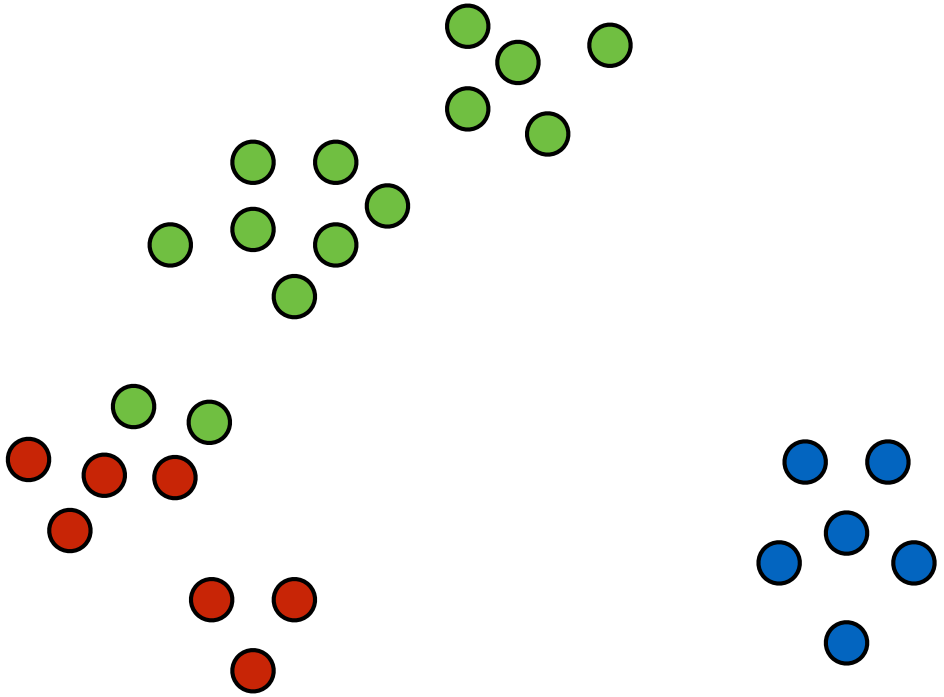
$$\mathrm{dist}(x, y) = \sum (x(i) - y(i))^2$$

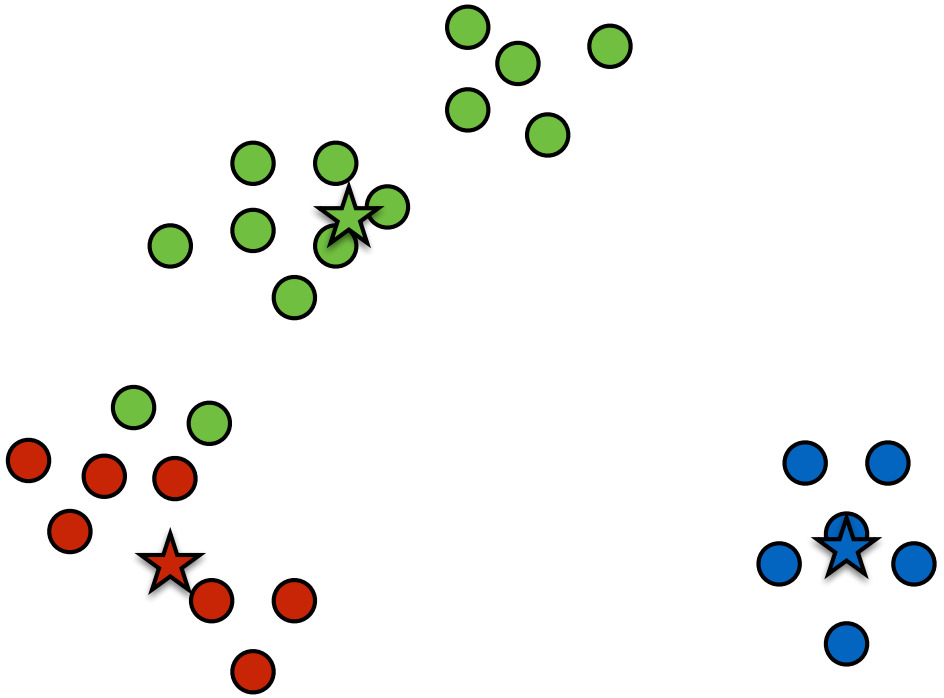
  - for each cluster, compute its new centroid (the mean of its vectors). $O(nd)$

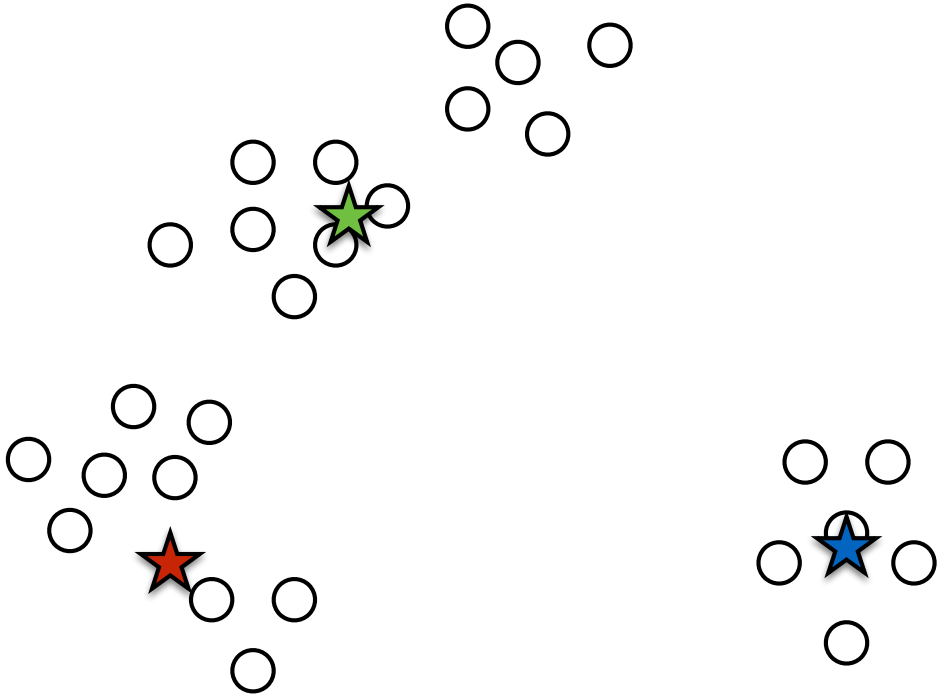


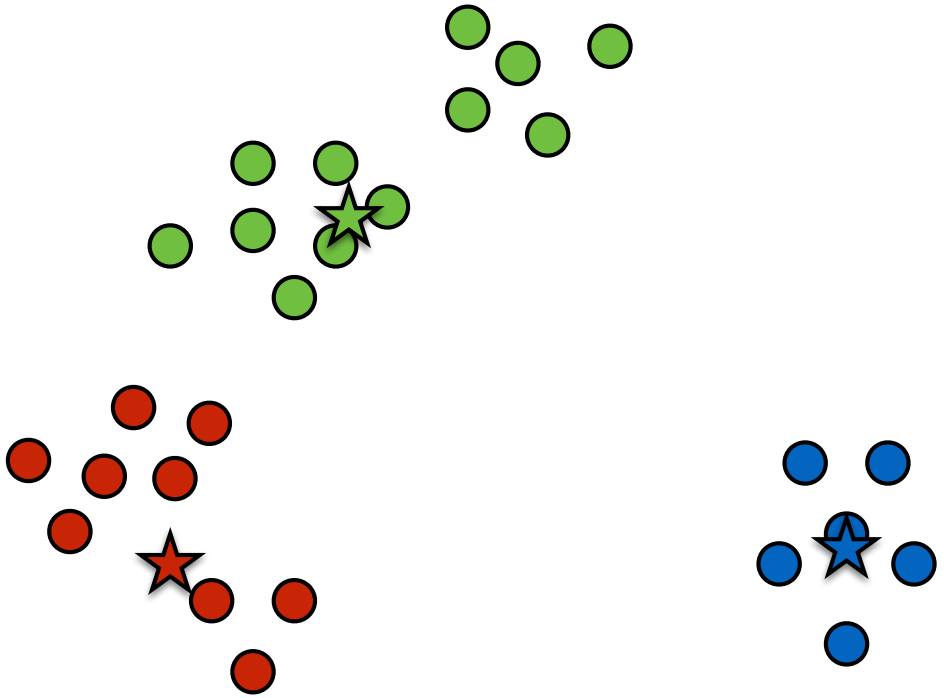


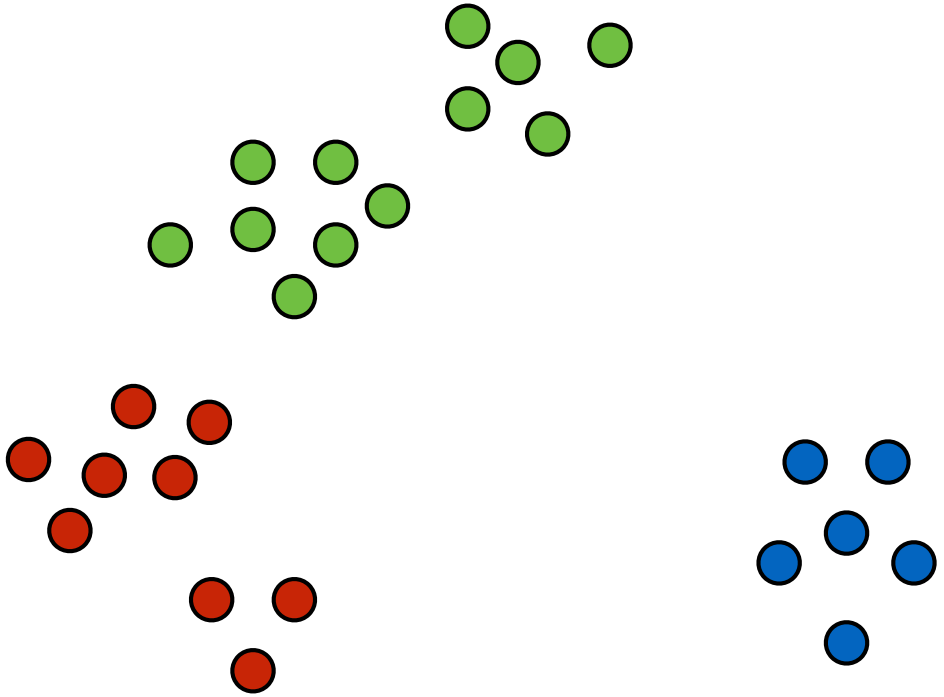


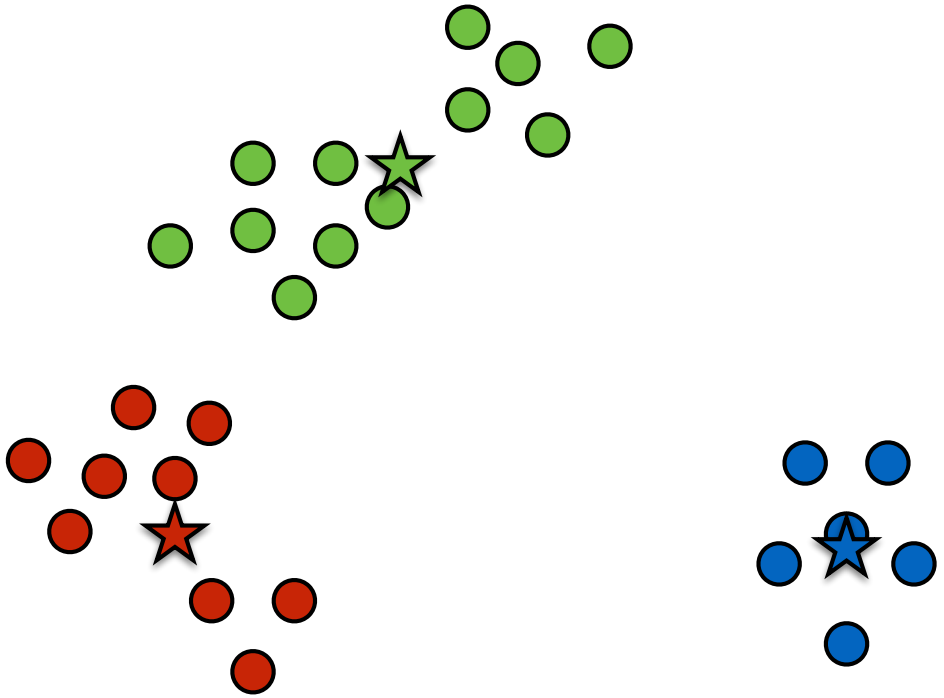


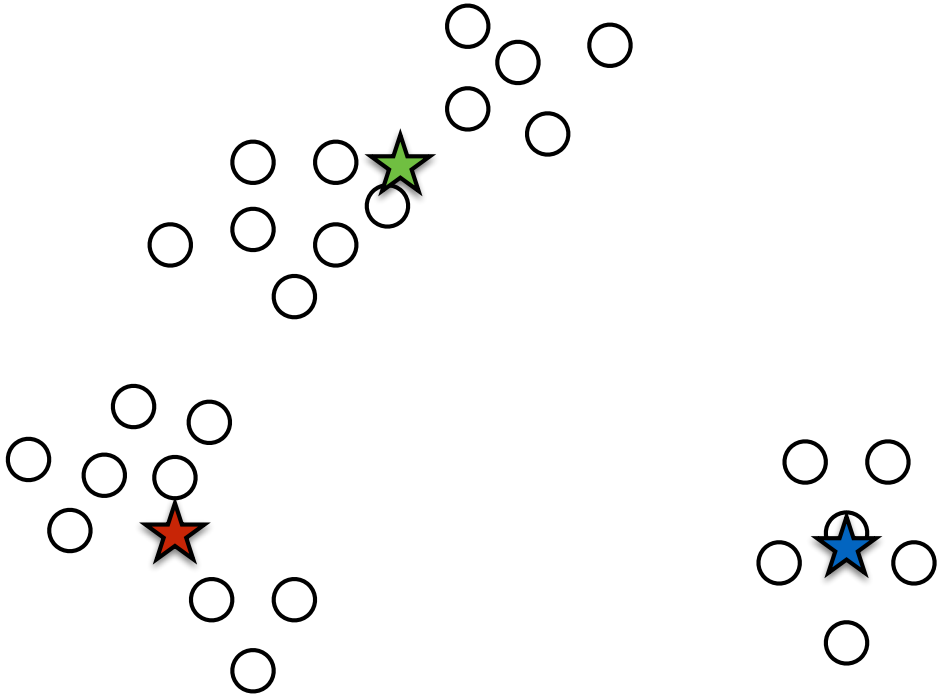


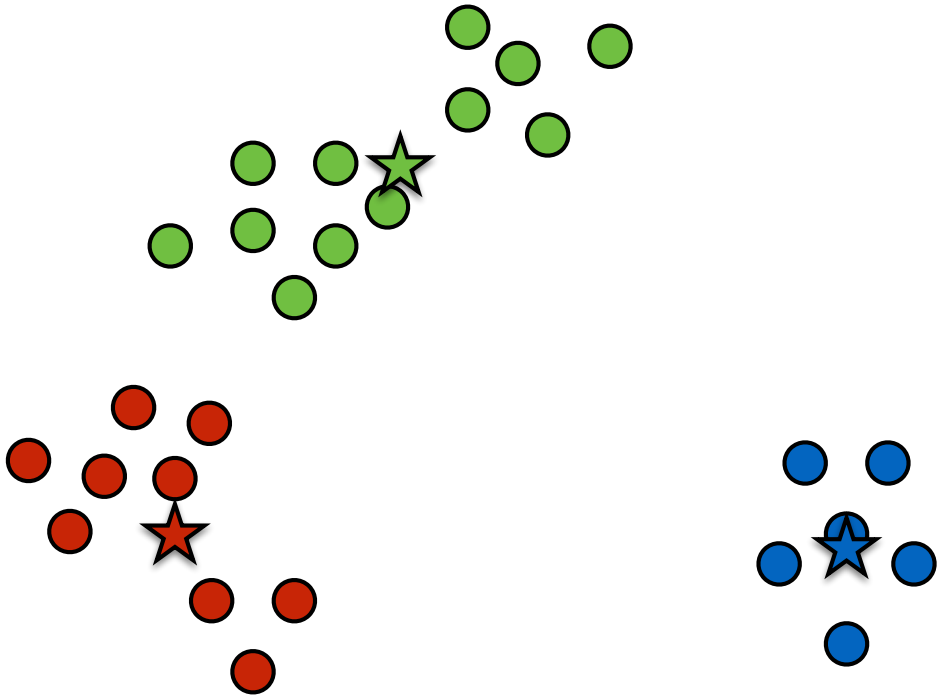












The clustering is finally stable

K-Means/Lloyd's Algorithm

- Does the algorithm computes the optimal k-means clustering? NO!
- Does it come close to it? NO! It can be arbitrarily bad
- Is the algorithm efficient? No! It can take **exponential time**
- What does it do? It computes a local minimum
- Why do people use it? Good question

K-Means Clustering

- Ok, Lloyd's algorithm does not compute the optimum..
- ..is there an **efficient** algorithm that does? No, and in fact, it **may not exist**

TIME IS OF THE ESSENCE

Time is of the essence

	N=50	N=100	N=1000	$N = 10^6$	$N = 10^9$
N	~0	~0	~0	0.001 sec	1 sec
N^2	~0	~0	0.001 sec	16 min	31 years
2^N	13 days	2871 U	$>10^{283}$ U	!	!
3^N	227,643 centuries	$> 10^{21}$ U	!	!	!
N!	$> 10^{46}$ U	$> 10^{140}$ U	!	!	!

computer speed = 1 billion operations per second

U is the Age of the Universe

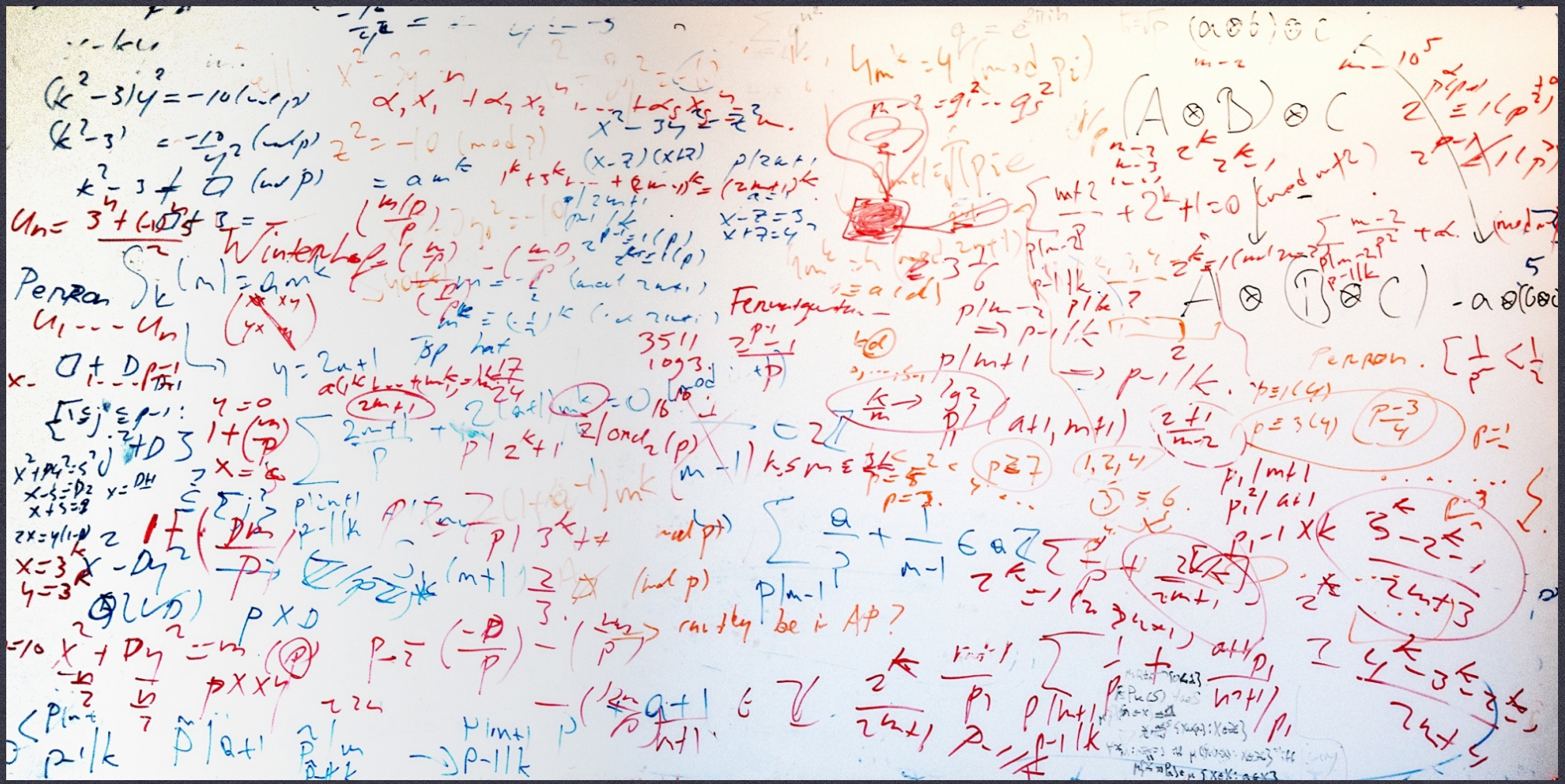
Time is of the essence

	N=50	N=100	N=1000	$N = 10^6$	$N = 10^9$
N	~0	~0	~0	~0	0.001 sec
N^2	~0	~0	~0	~0	11 days
2^N	18 minutes	2.8 U	$>10^{280}$ U	!	!
3^N	226 centuries	$> 10^{18}$ U	!	!	!
N!	$> 10^{43}$ U	$> 10^{137}$ U	!	!	!

computer speed improves one thousand times
U is the Age of the Universe

Moral of the story

- No technological improvement will ever make an exponential-time algorithm practical



BACK TO THE BOARD