Clustering & the K-means algorithm

Announcements:

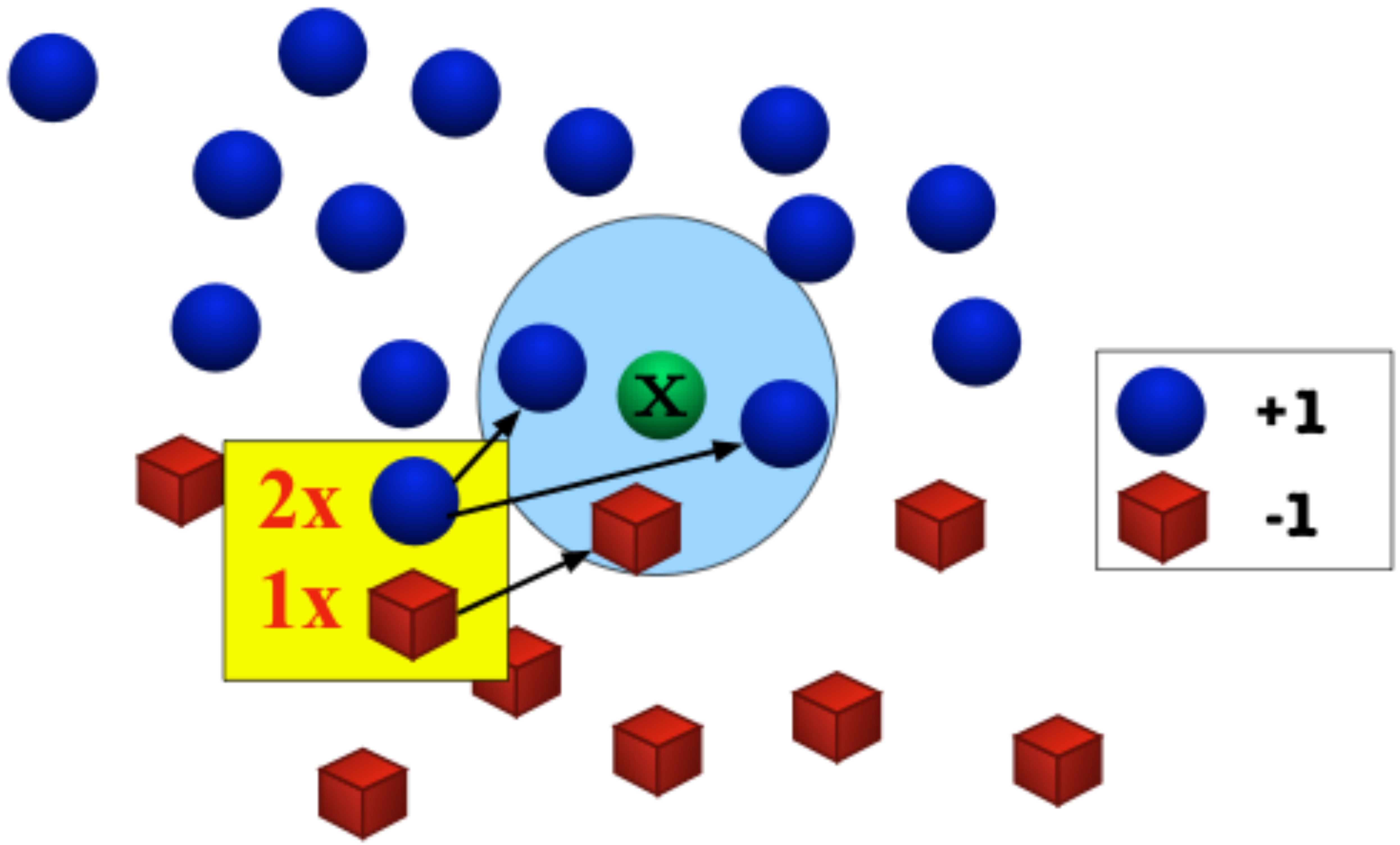
1. HW1 is out, due Sep 12

2. P1 will be out this afternoon

3. CIS partner finding social: this Friday 4-6, Gates 01

Recap

The K-NN algorithm



Example: 3-NN with Euclidean distance on a binary classification data

Recap

T/F: We can use train-validation trick to determine the parameter K

T/F: in worst case, number of training example should scale in $\exp(d)$ for K-NN to succeed

T/F: K-NN will fail when feature dimension is high

Objective

Understand the K-means algorithm and why it works

Outline for Today

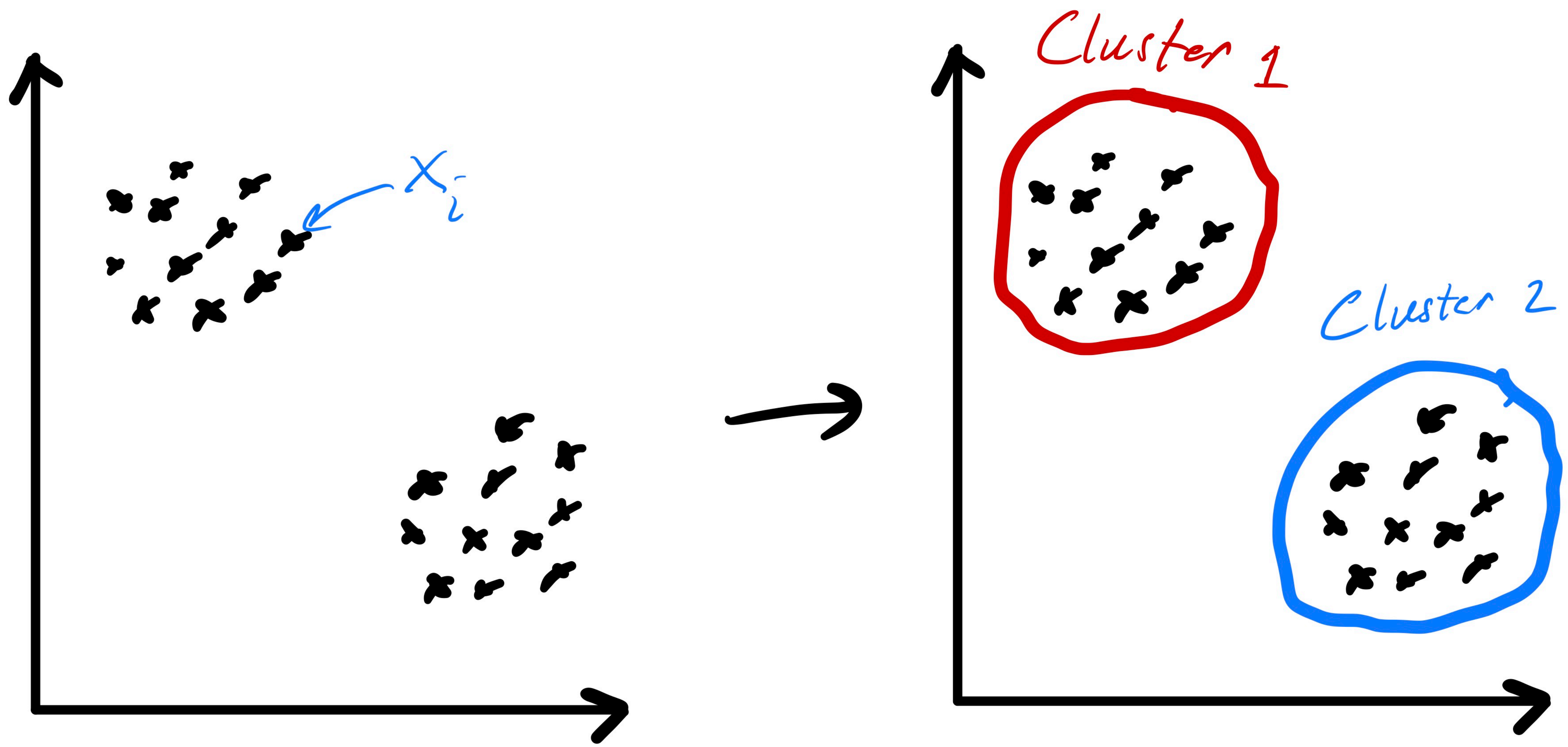
1. Unsupervised Learning: Clustering

2 the K-means algorithm

3. Convergence of K-means

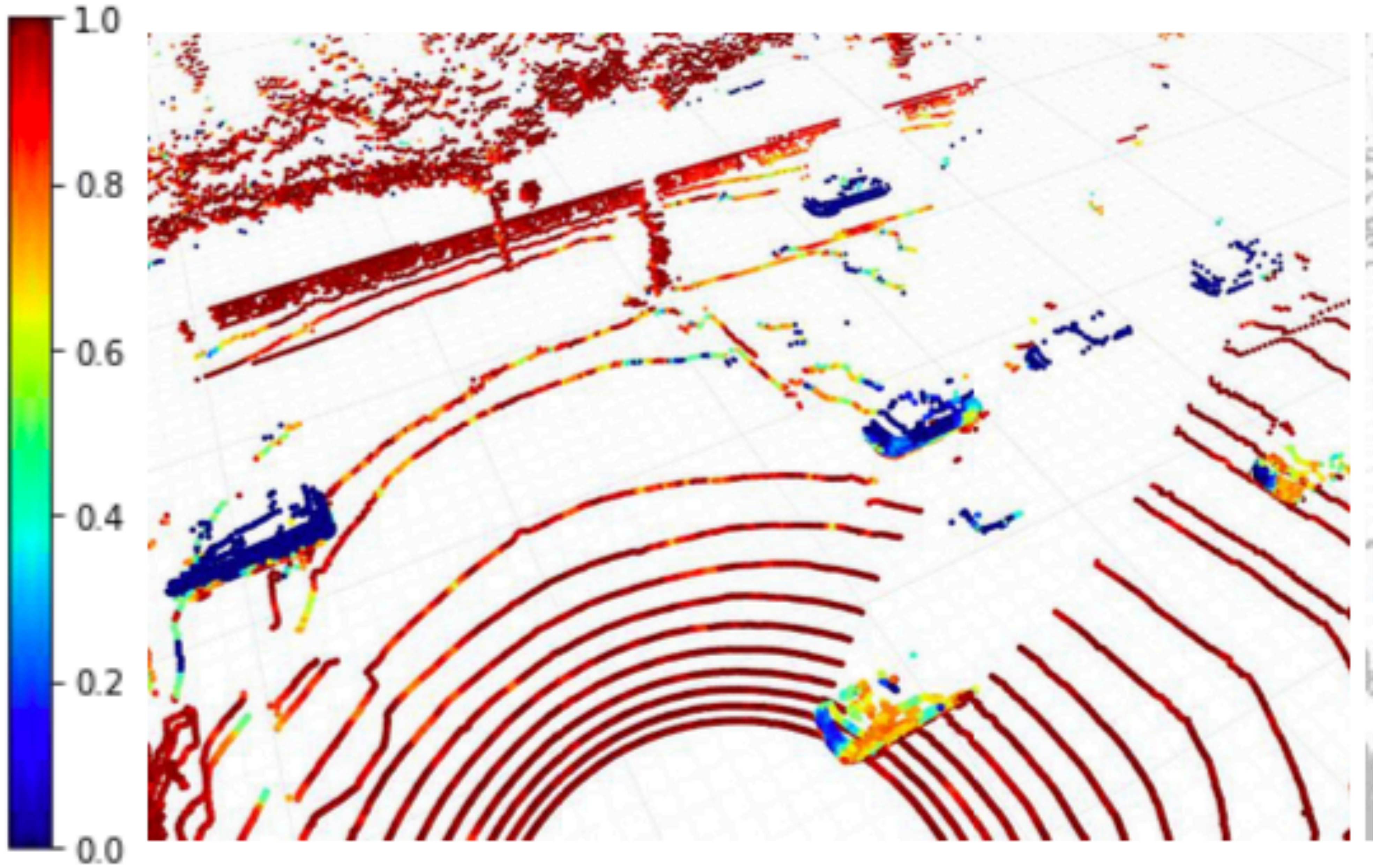
What is clustering?

It is an unsupervised learning procedure (i.e., applies to data without ground truth labels)

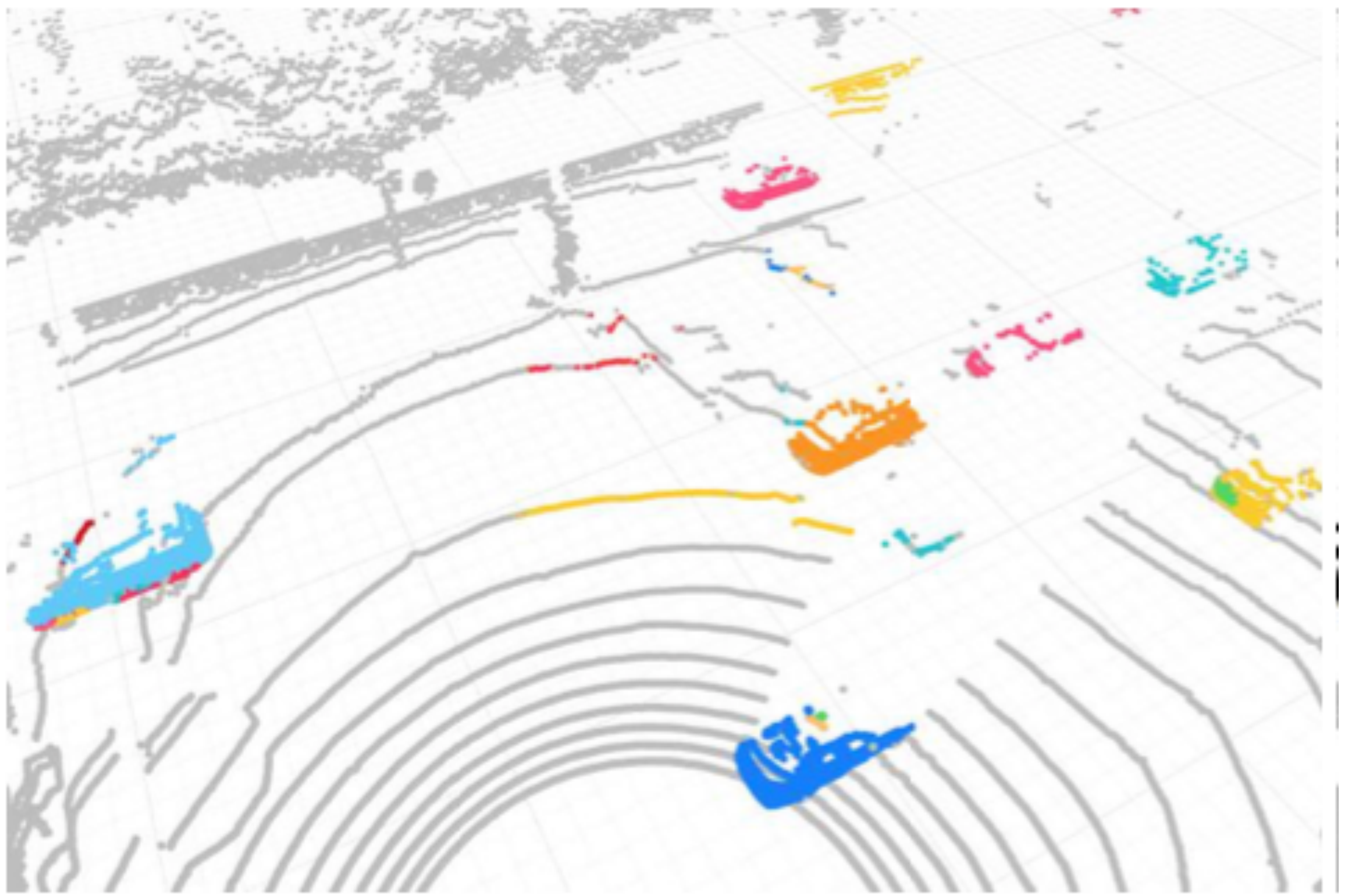


Usage of clustering algorithms in real world

Example: Learning to detect cars without ground truth label



Apply clustering on this dataset (point cloud)



A point cloud from a Lidar sweep (4-d data)

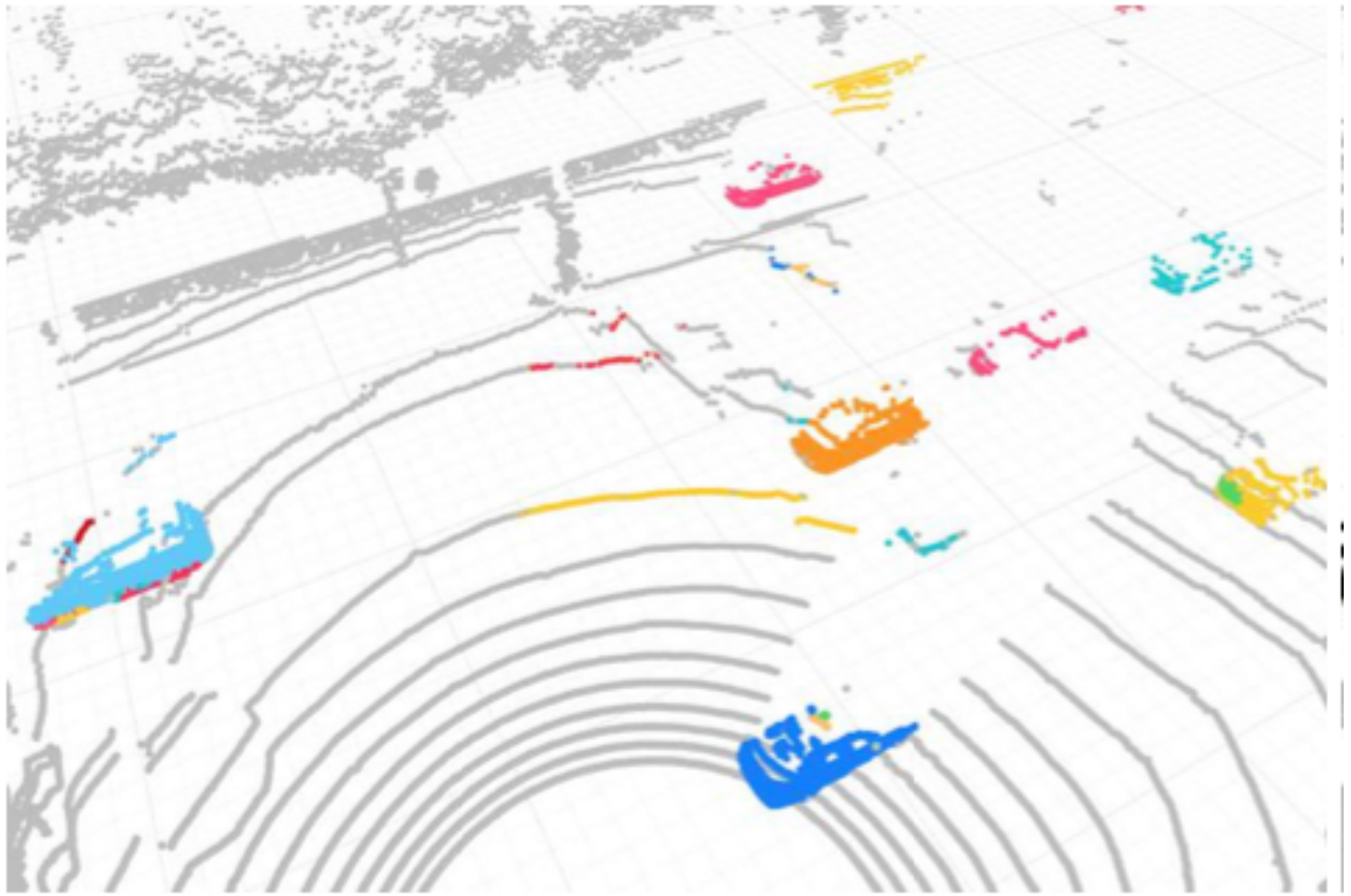
Different color represents different clusters

Usage of clustering algorithms in real world

Example: Learning to detect cars without ground truth label

3. Fit Bounding Boxes

Fitting bounding box around clusters

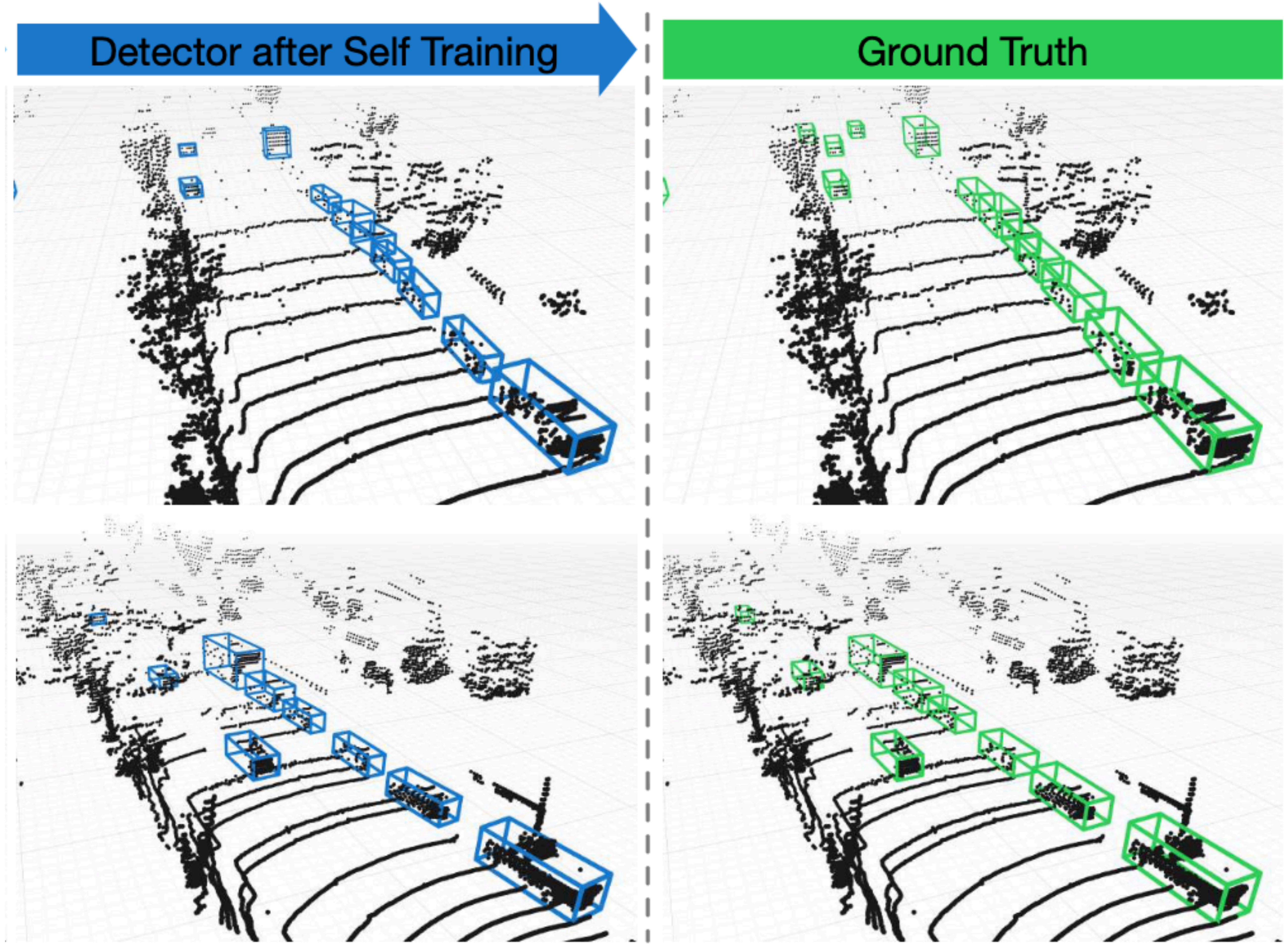


These boxes are the pseudo-labels we use to train detector

Different color represents different clusters

Usage of clustering algorithms in real world

Example: Learning to detect cars without ground truth label



Outline for Today

1. Unsupervised Learning: Clustering

2 the K-means algorithm

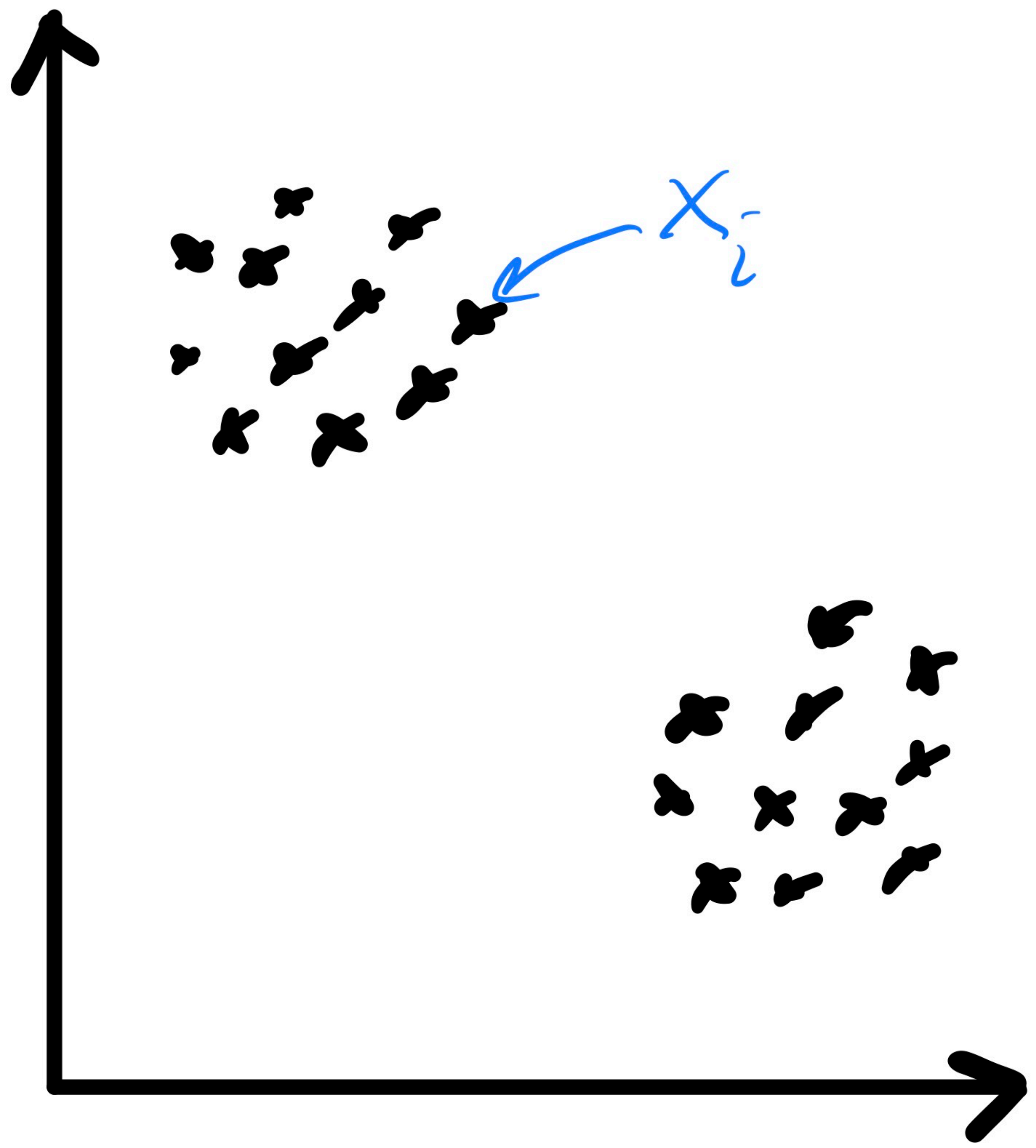
3. Convergence of K-means

The K-means algorithm

Input $\mathscr{D} = \{x_1, \ldots, x_n\}, x_i \in \mathbb{R}^d$, parameters K

Expected output: K centroids $\{\mu_1, \mu_2, \ldots, \mu_k\}, \mu_i \in \mathbb{R}^d$, and K clusters $C_1 \ldots, C_K$

The data assignment procedure:

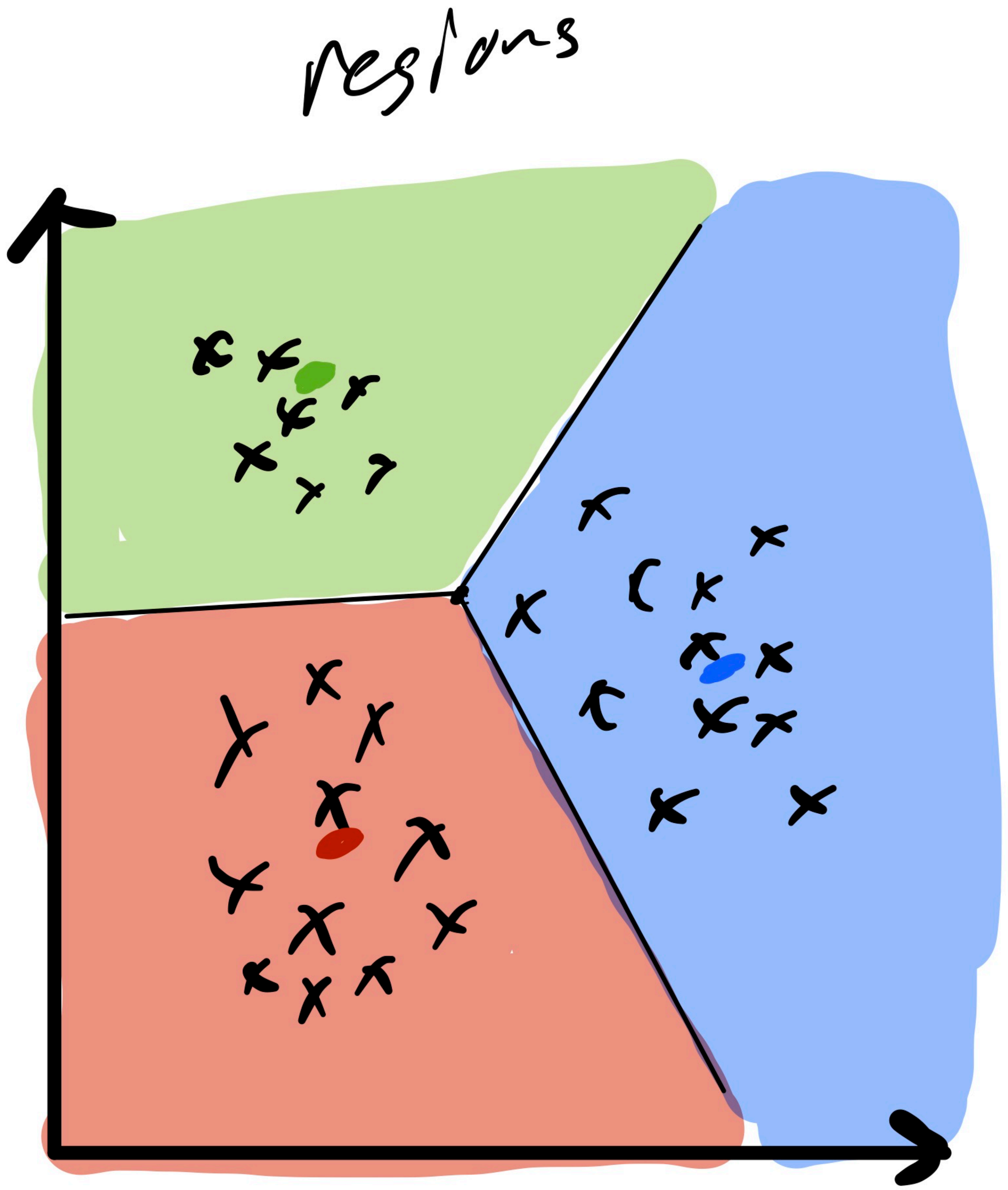


<u>If we had K centroids,</u> we could split the dataset into K clusters, $C_1, \ldots, C_K$, by assigning each data point to its nearest centroid

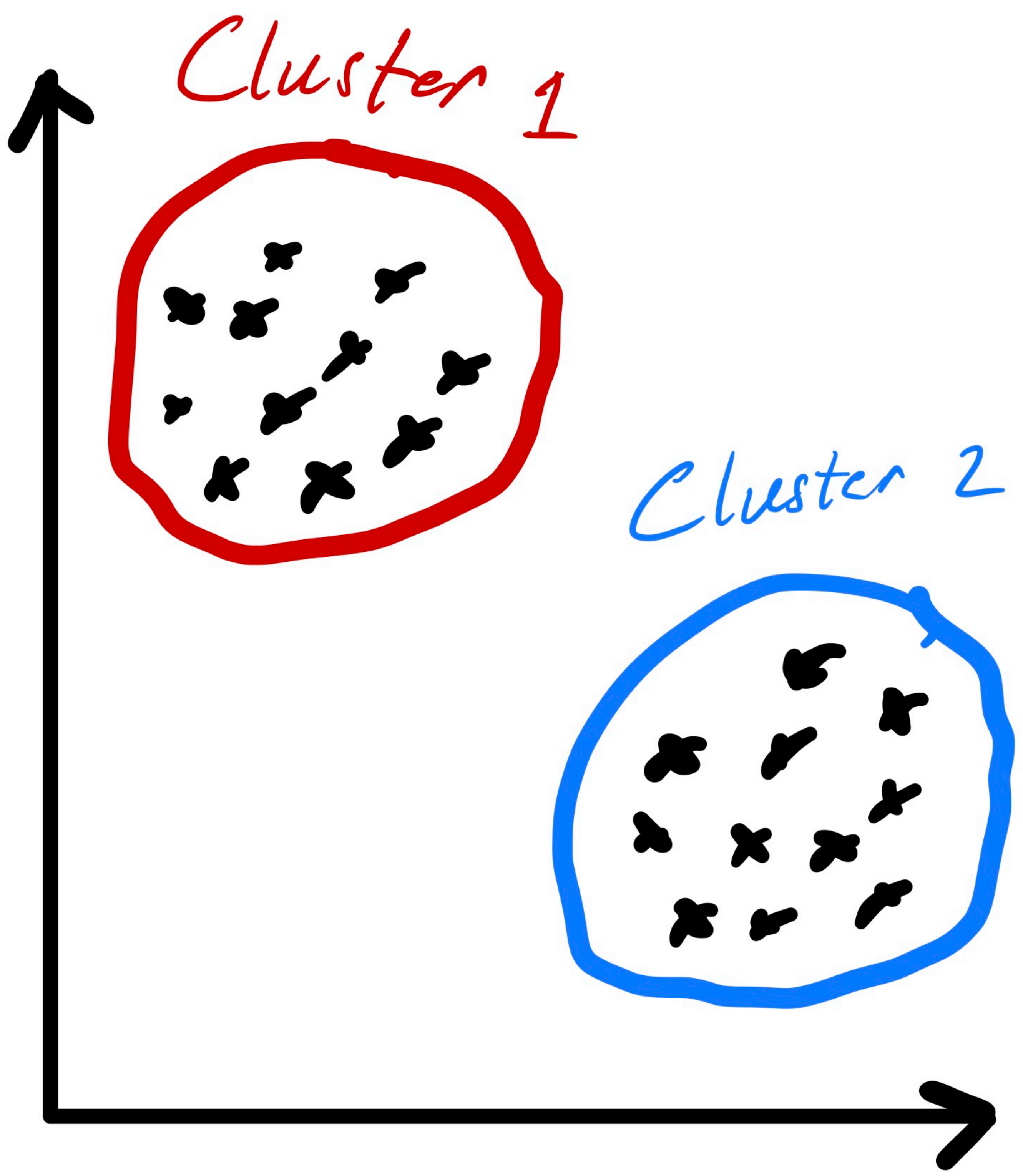
$$C_i = \{x \in \mathscr{D} \text{ s.t., } \mu_i \text{ is the closest centroid to } x\}$$

The data assignment procedure

K centroids $\mu_1, \ldots, \mu_k$ splits the space into a voronoi diagram



The centroid computation procedure



If we magically had the clusters $C_1, \ldots, C_K$, we could compute centroids as follows:

μ_i : the mean of the data in C_i

The K-means algorithm

Iterate between Centroid computation and Data Assignment!

Initialize K clusters $C_1, C_2, \ldots, C_K$, where $\cup_{i=1}^{K} C_i = \mathscr{D}$, and $C_i \cap C_j = \emptyset$, for $i \neq j$

Repeat until convergence:

1. centroids computation using $C_1, \ldots, C_K$, i.e.,for all i,
$\mu_i = \sum_{x \in C_i} x / |C_i|$ (i.e., the mean of the data in C_i)

2. the data assignment procedure, i.e., re-split data into $C_1, \ldots, C_K$, using $\mu_1, \ldots, \mu_k$

The K-means algorithm

Let’s try out K-means!

Outline for Today

1. Unsupervised Learning: Clustering

2 the K-means algorithm

3. Convergence of K-means

Does K-means algorithm converge?

Yes, though it does not guarantee to return the globally optimal solution

Given any K disjoint groups $C_1, C_2, \ldots, C_K$, and any K centroids, define a loss function:

$$\ell(\{C_i\}, \{\mu_i\}) = \sum_{i=1}^{K} \left[\sum_{x \in C_i} \|x - \mu_i\|_2^2 \right]$$

Total distance of points in C_i to μ_i

K-means as a Coordinate Descent Algorithm

$$\ell(\{C_i\}, \{\mu_i\}) = \sum_{i=1}^{K} \left[\sum_{x \in C_i} \|x - \mu_i\|_2^2 \right]$$

K-means minimizes ℓ in an alternating fashion:

Q1: w/ $C_1, \ldots, C_K$ fix, what is $\arg \min_{\mu_1, \ldots, \mu_k} \ell(\{C_i\}, \{\mu_i\})$?

Q2: w/ $\mu_1, \ldots, \mu_K$ fix, what is $\arg \min_{C_1, \ldots, C_k} \ell(\{C_i\}, \{\mu_i\})$?

K means is doing Coordinate Descent here

$$\ell(\{C_i\}, \{\mu_i\}) = \sum_{i=1}^{K} \left[\sum_{x \in C_i} \|x - \mu_i\|_2^2 \right]$$

K-means Algorithm: (re-stated from a different perspective)

Initialize $\mu_1, \ldots, \mu_K$

Repeat until convergence:

$$C_1, \ldots, C_K = \arg \min_{C_1, \ldots, C_k} \ell(\{C_i\}, \{\mu_i\})$$

$$\mu_1, \ldots, \mu_K = \arg \min_{\mu_1, \ldots, \mu_k} \ell(\{C_i\}, \{\mu_i\})$$

How to pick K?

Given K, we can look at the minimum loss

$$\ell_K := \min_{C_1,\ldots,C_K,\mu_1,\ldots,\mu_K} \ell(\{C_i\}, \{\mu_i\}) = \sum_{i=1}^{K} \left[\sum_{x \in C_i} \|x - \mu_i\|_2^2 \right]$$

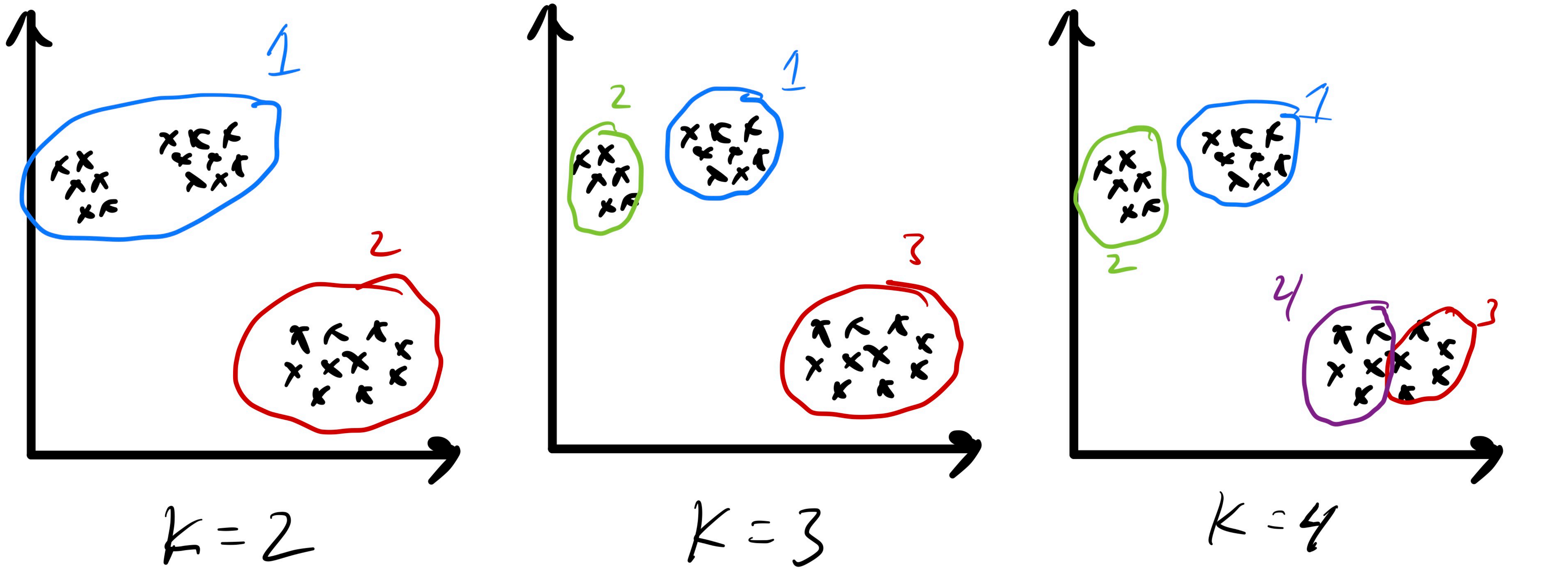
Note that exactly compute the $\min$ is NP-hard, but we can approximate it w/ K-means solutions

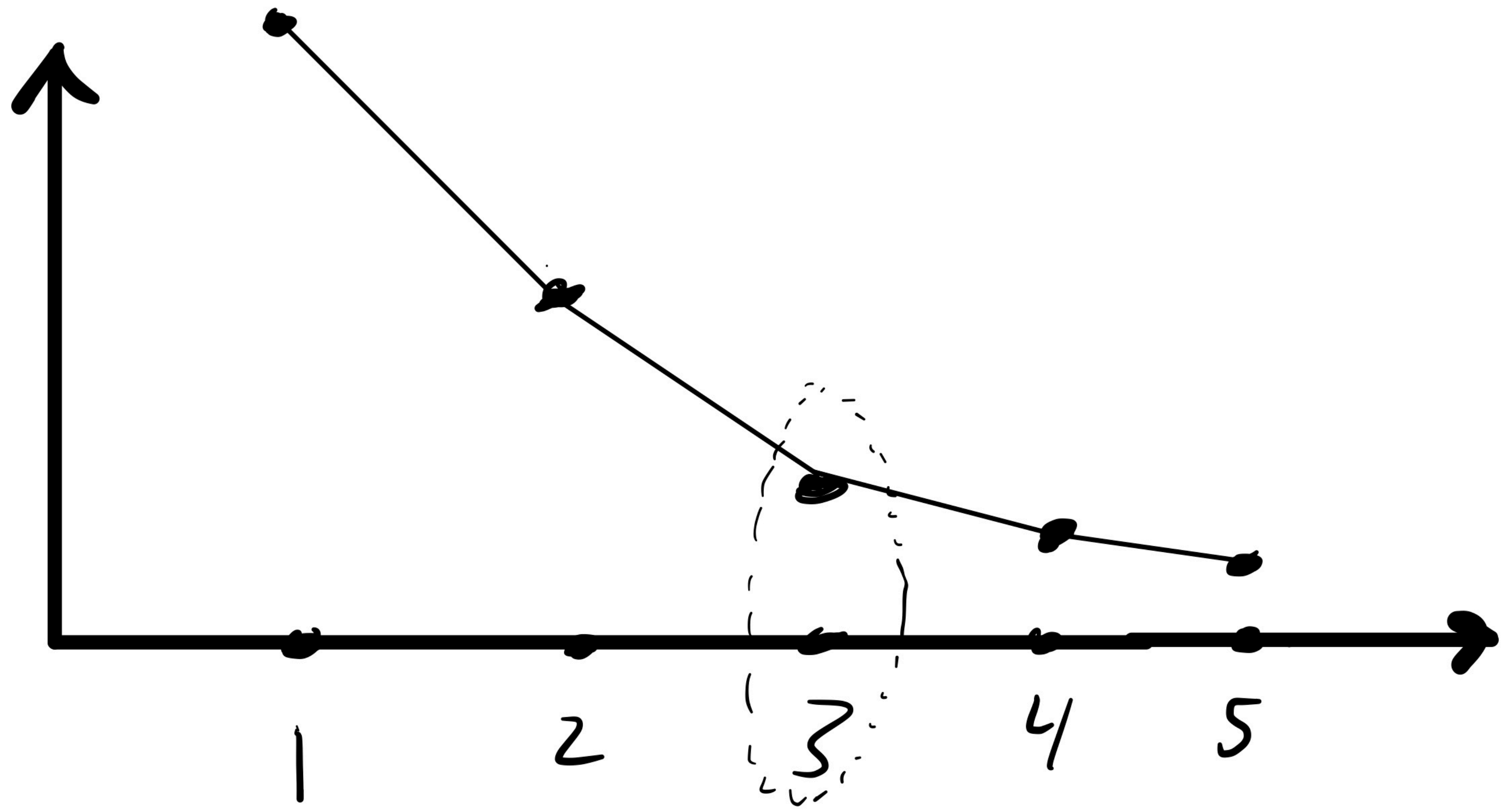
Q: Should we just naively pick a K that the ℓ_K is zero?

No! When $K = n$, loss is zero (every data point is a cluster!)

How to pick K?

In practice, we can gradually increase K, and keep track the loss ℓ_K, and stop when ℓ_K does not drop too much





Summary

1. The first Unsupervised Learning Algorithm — K means

iterratively computes centroids and clusters

2. Relationship between K-means algorithm and the Coordinate descent procedure on loss $\ell(\{C_i\}, \{\mu_i\})$