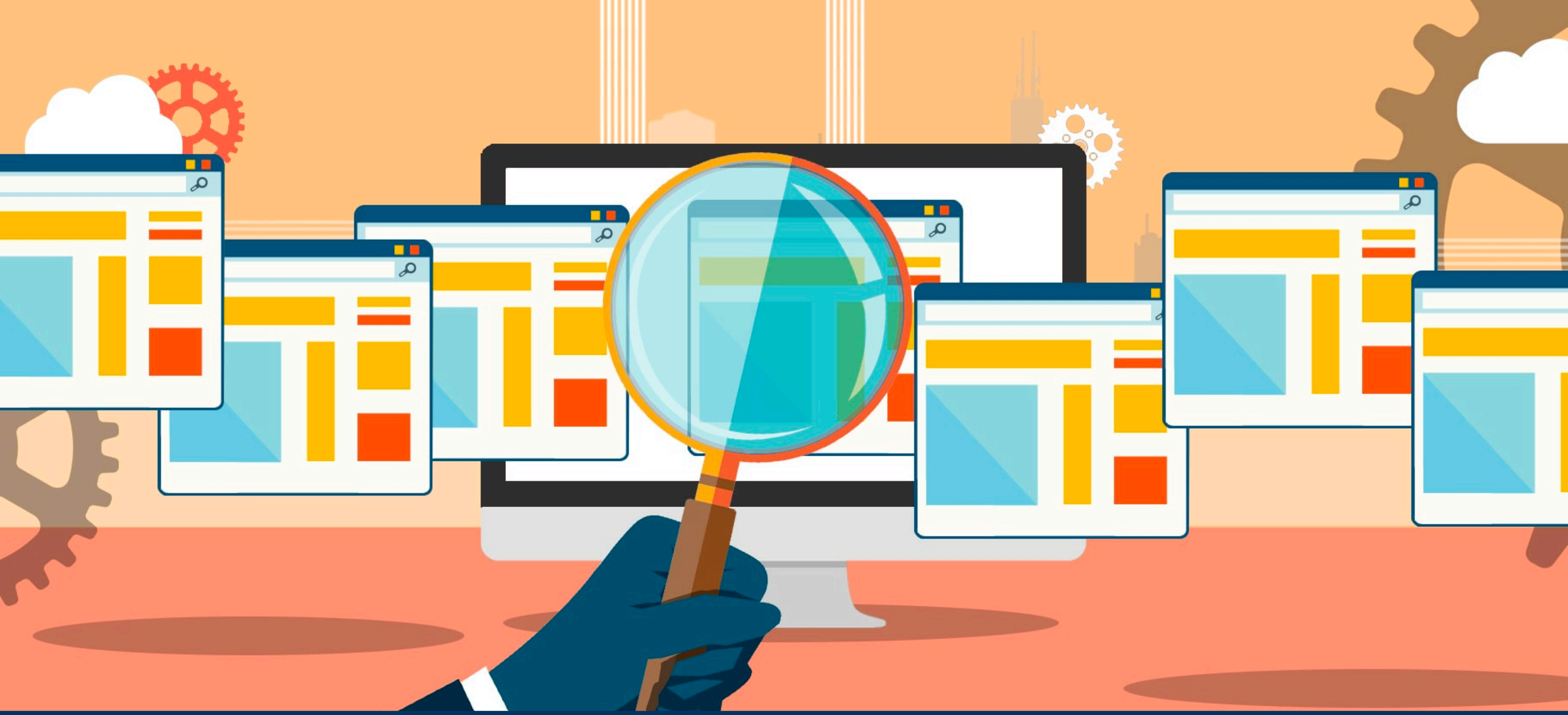


Clustering Continued

INST414 - Data Science Techniques

This Module's Learning Objectives

Last Time

Differentiate between unsupervised and supervised machine learning

Formally define "clustering"

Describe how one represents a cluster of multiple points

Explain how one can calculate distances between clusters

This Module's Learning Objectives

Last Time

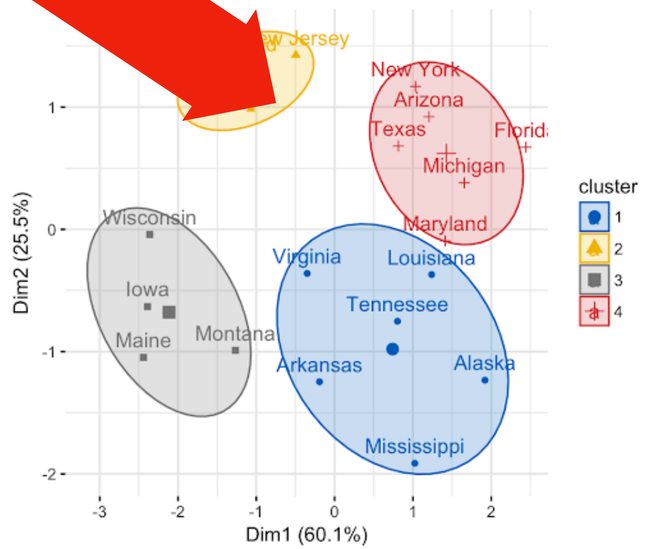
Differentiate between unsupervised and supervised machine learning

Formally define "clustering"

Describe how one represents a cluster of multiple points

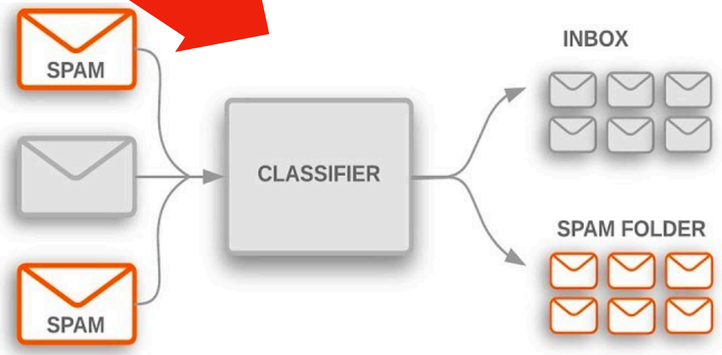
Explain how one can calculate distances between clusters

Unsupervised



"Learn" some structure in the data

Supervised



"Learn" to generalize from examples of a task

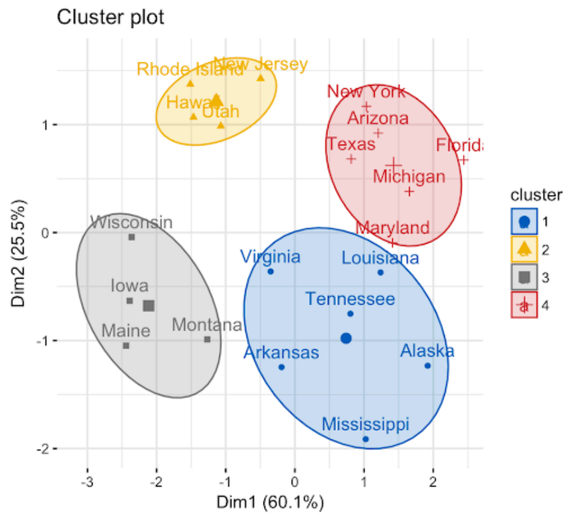
Unsupervised:
You want to find structure in your data, but you don't have examples of this structure

How do I know if the problem is "unsupervised" or "supervised"?

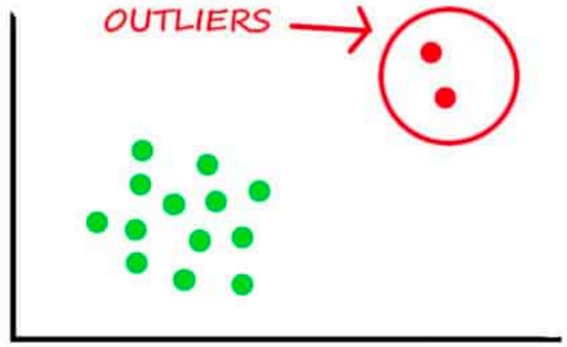
Supervised:
You want to recover (predict) <u>known relationships</u> in your data, and you have examples (i.e., labels) of these relations

Do you have "Labels"?

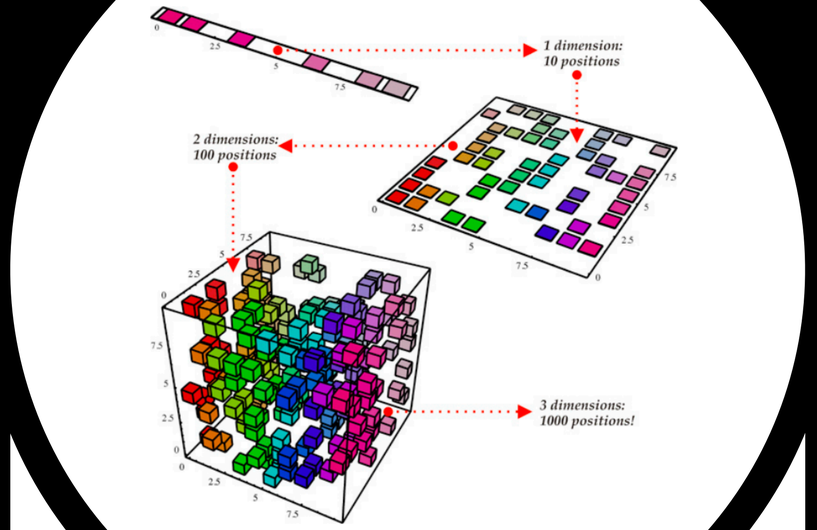
Unsupervised Learning



Extracting “similar” elements (clustering)

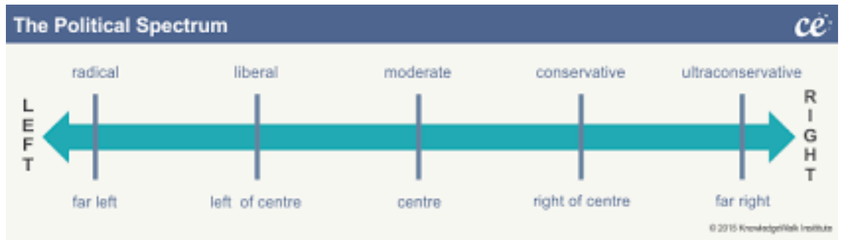


Identifying “dissimilar” elements (outliers)

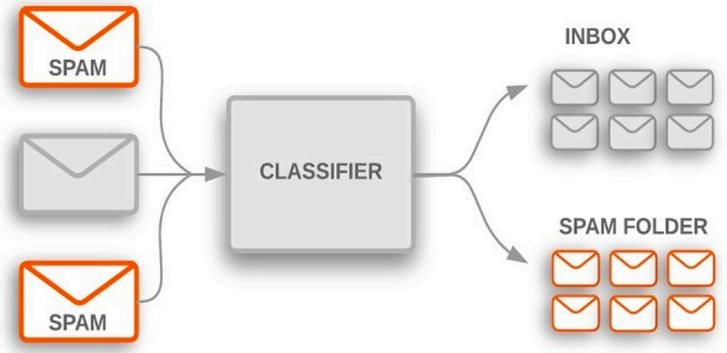


Extracting latent structure (e.g., dim. reduction)

Supervised Learning



Predicting scores based on known, continuous labels (regression)



Classifying elements based on known, categorical labels (classification)

So how might we cluster in given space?

This Module's Learning Objectives

Last Time

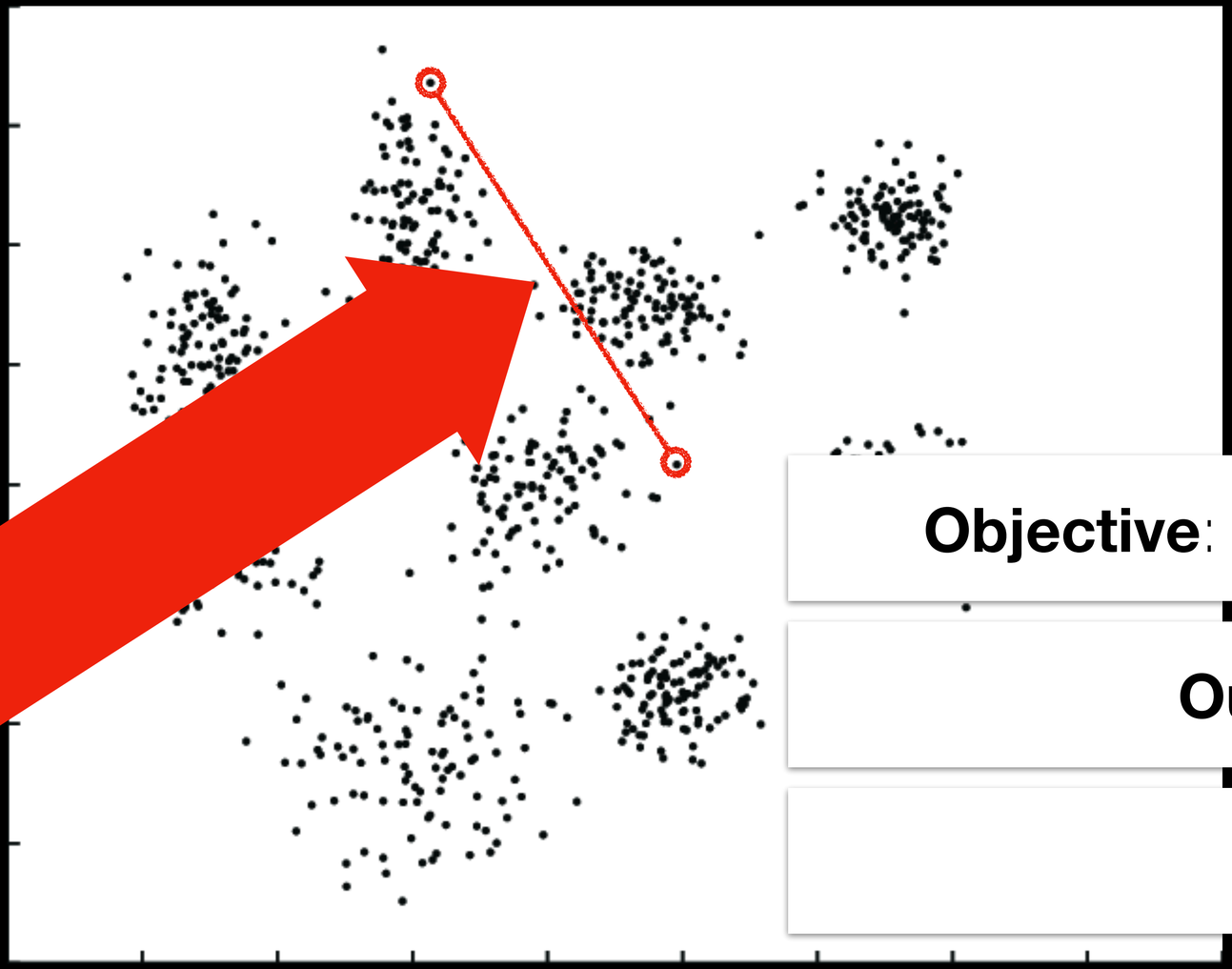
Differentiate between unsupervised and supervised machine learning

Formally define "clustering"

Describe how one represents a cluster of multiple points

Explain how one can calculate distances between clusters

A Formal Definition for Clustering



Given a **set of points**…

And a “**distance**” among them

Objective: Group points into ***k*** clusters, such that…

Output: List of assignments for each point

Of the form <data point, cluster id>

How many clusters are there in this image?

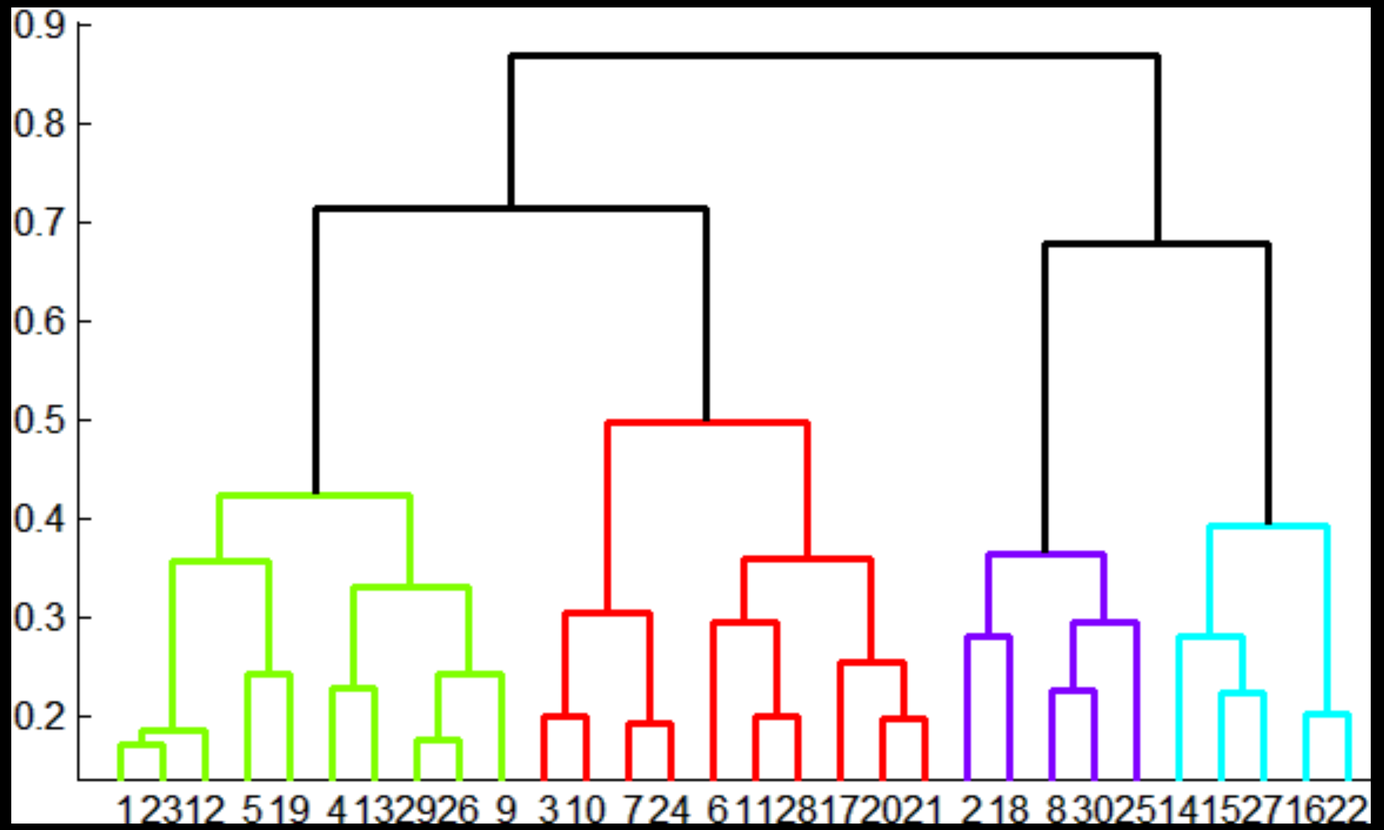
Cluster Assignments

Point	Cluster
d1	0
d2	0
d3	0
…	…
dn	10

What's should clustering output be here?

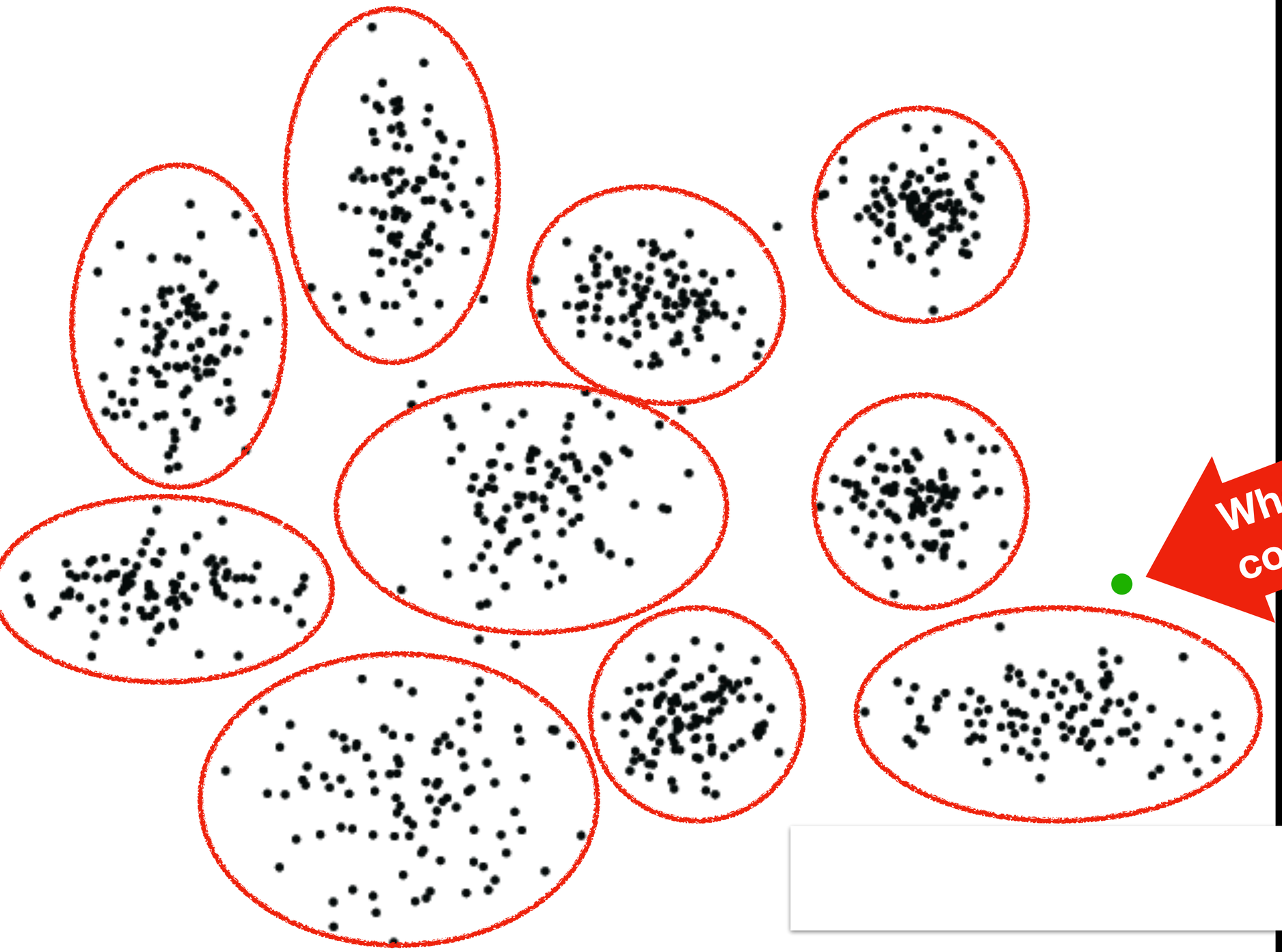
Two main ways to do clustering:

Hierarchical Clustering



Agglomerative

Divisive



Point Assignment

This Module's Learning Objectives

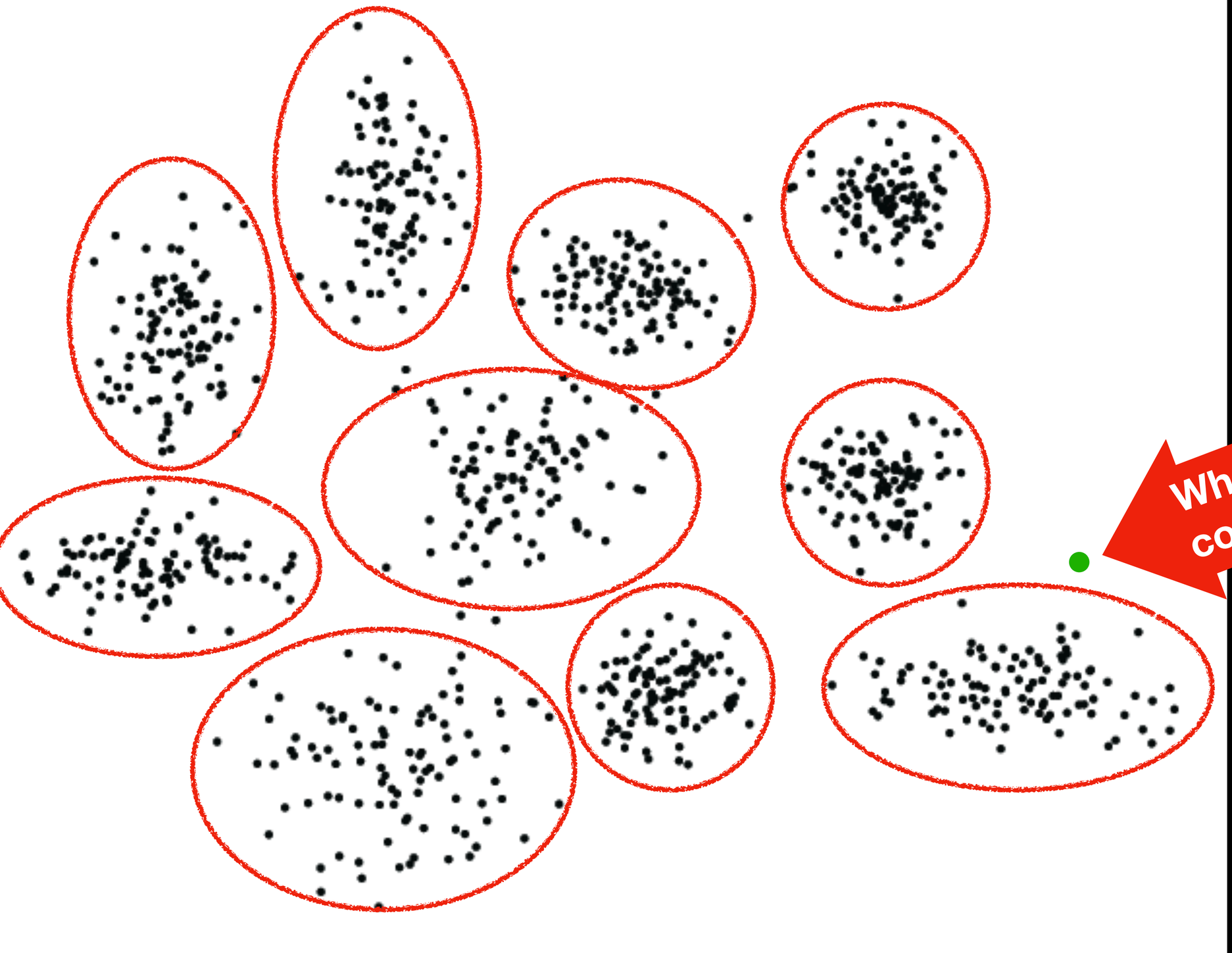
Last Time

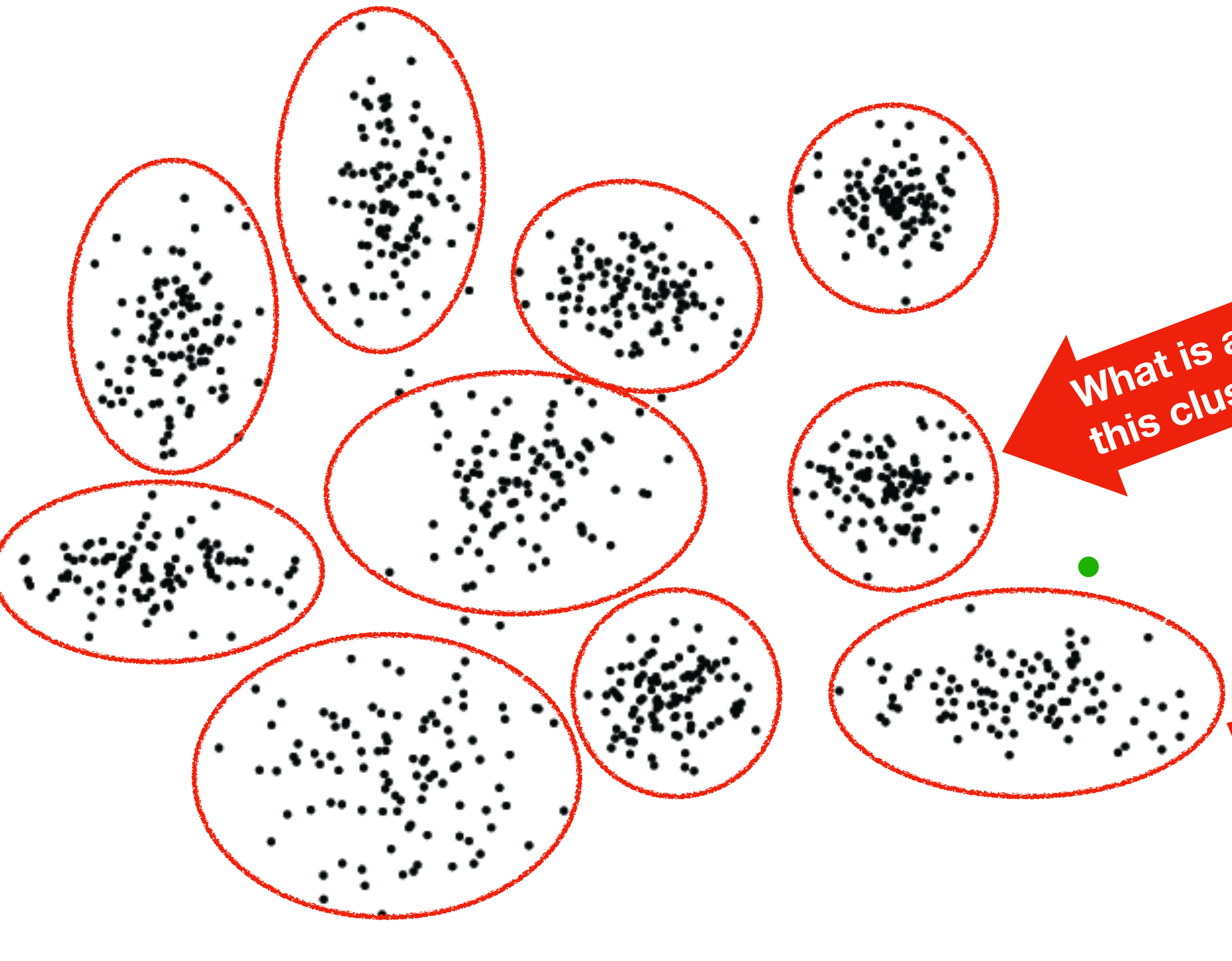
Differentiate between unsupervised and supervised machine learning

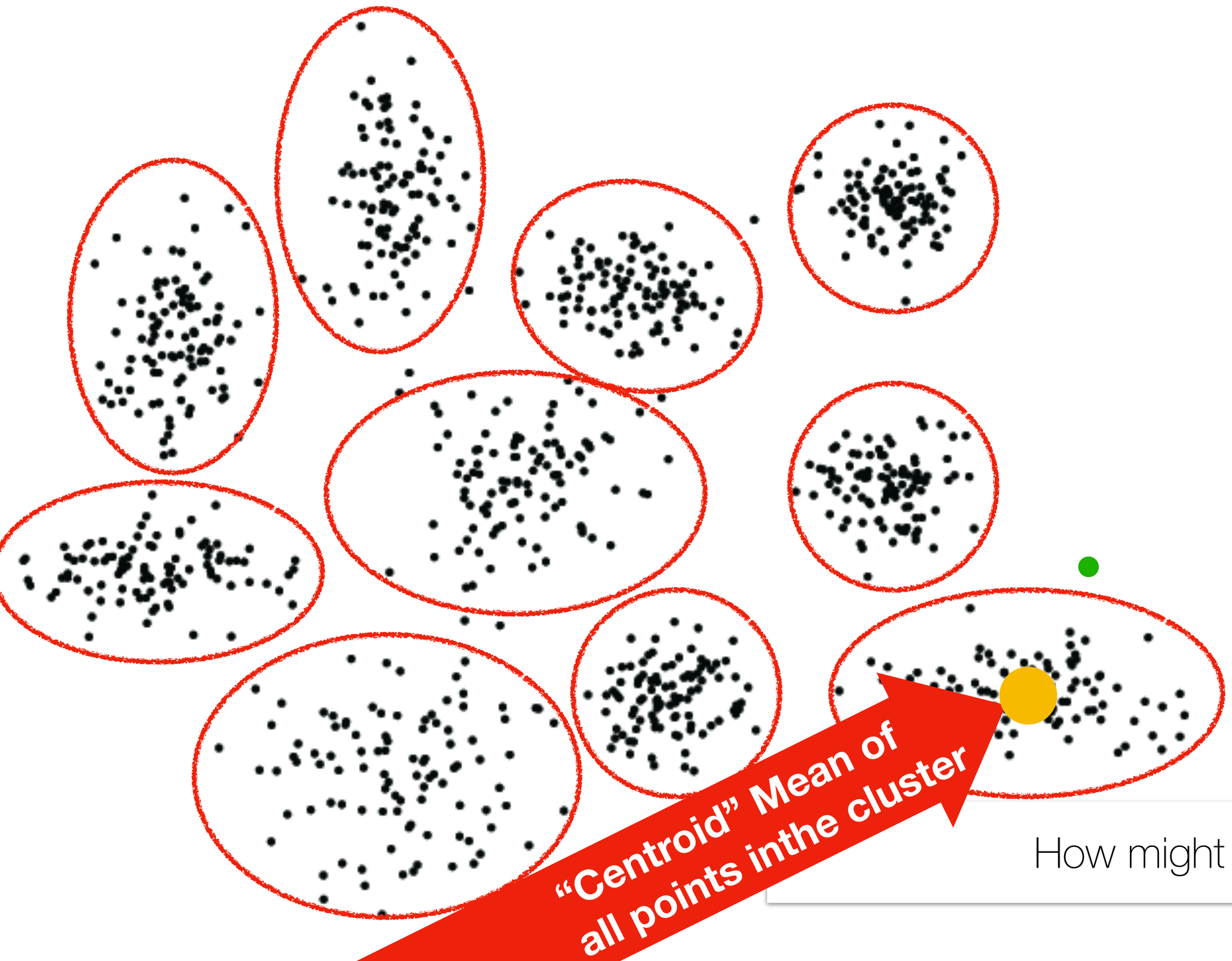
Formally define "clustering"

Describe how one represents a cluster of multiple points

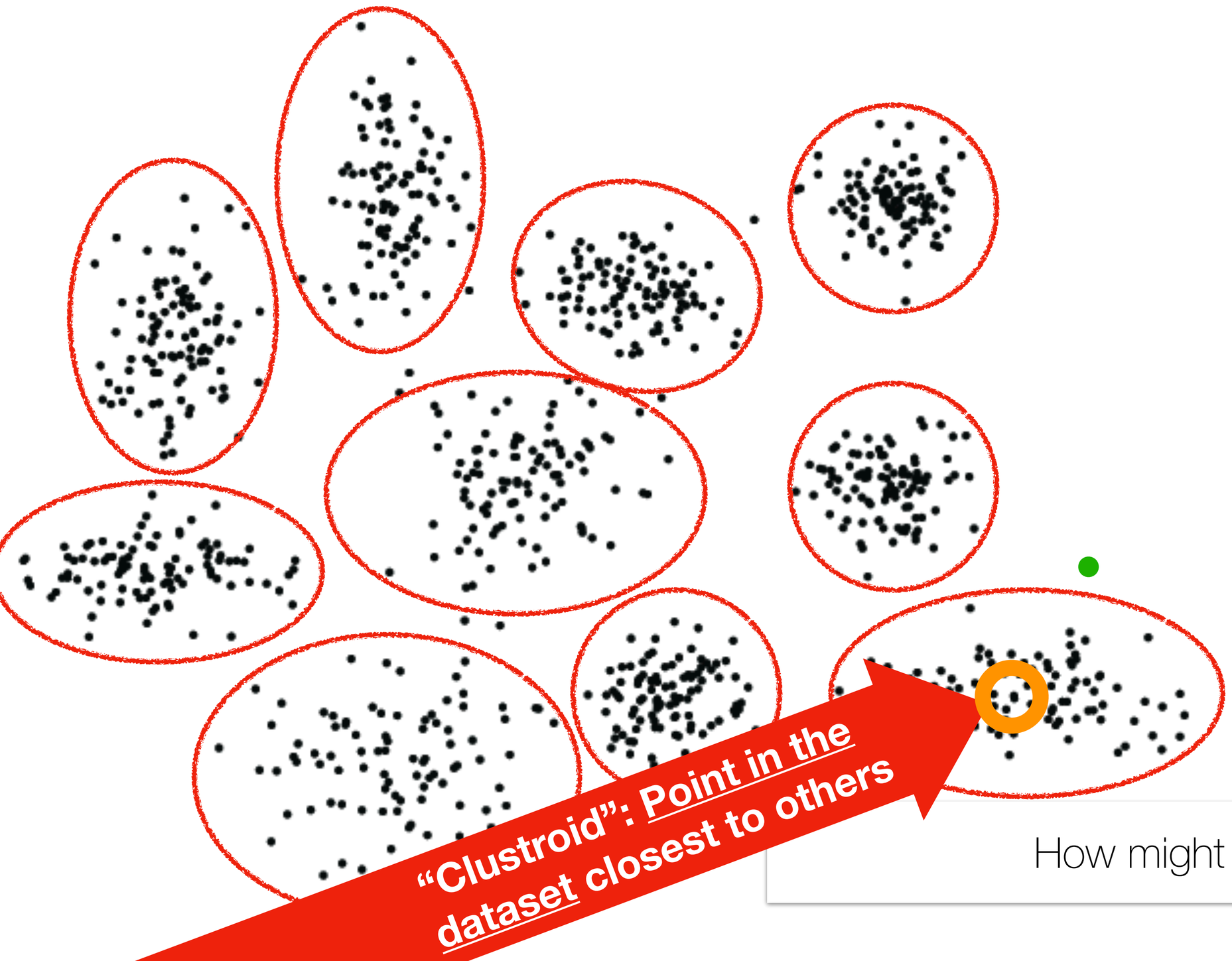
Explain how one can calculate distances between clusters


What cluster should
contain this point


What is are the coordinates of this cluster?
What is are the coordin this cluster?



How might we represent this cluster?



How might we represent this cluster?

This Module's Learning Objectives

Last Time

Differentiate between unsupervised and supervised machine learning

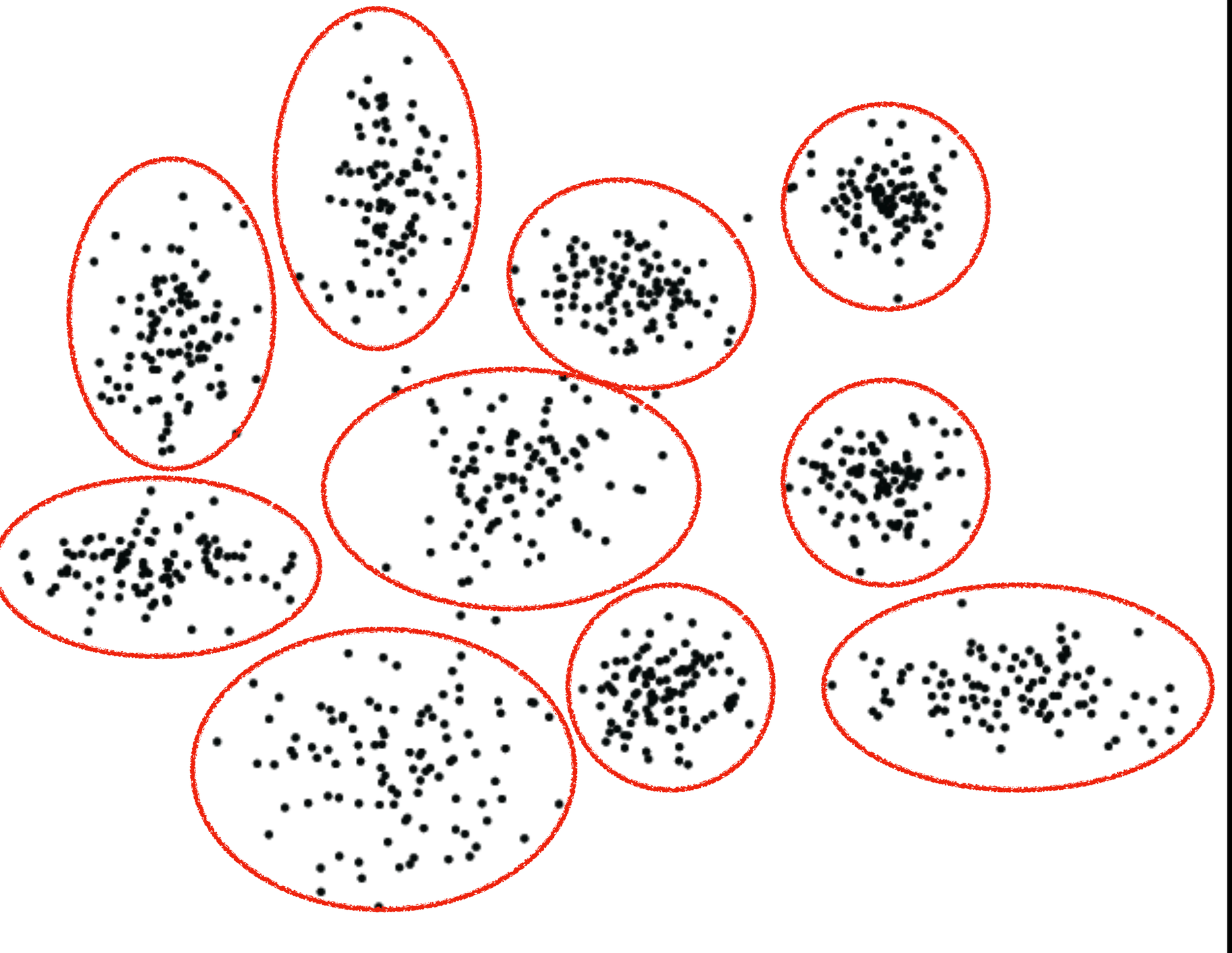
Formally define "clustering"

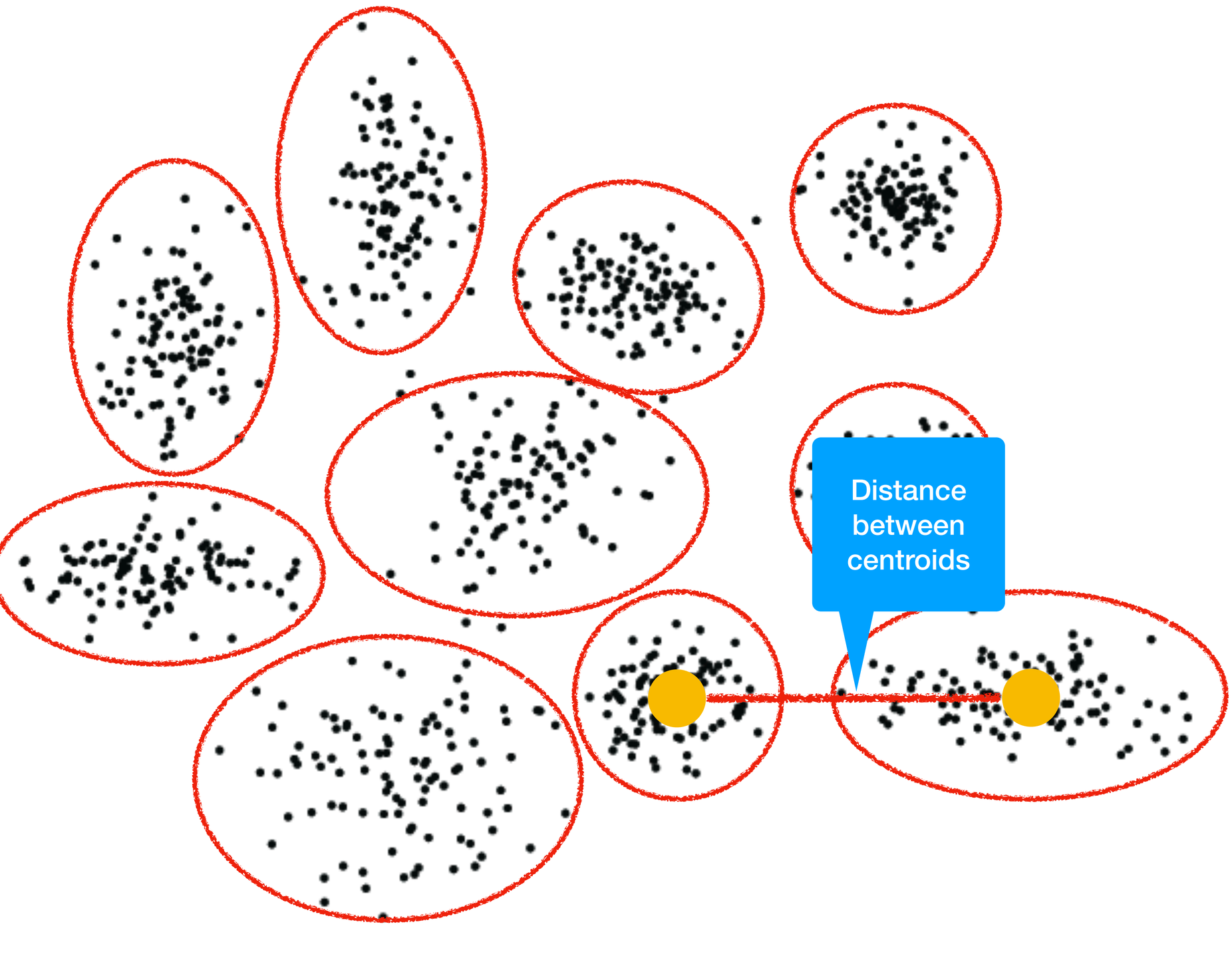
Describe how one represents a cluster of multiple points

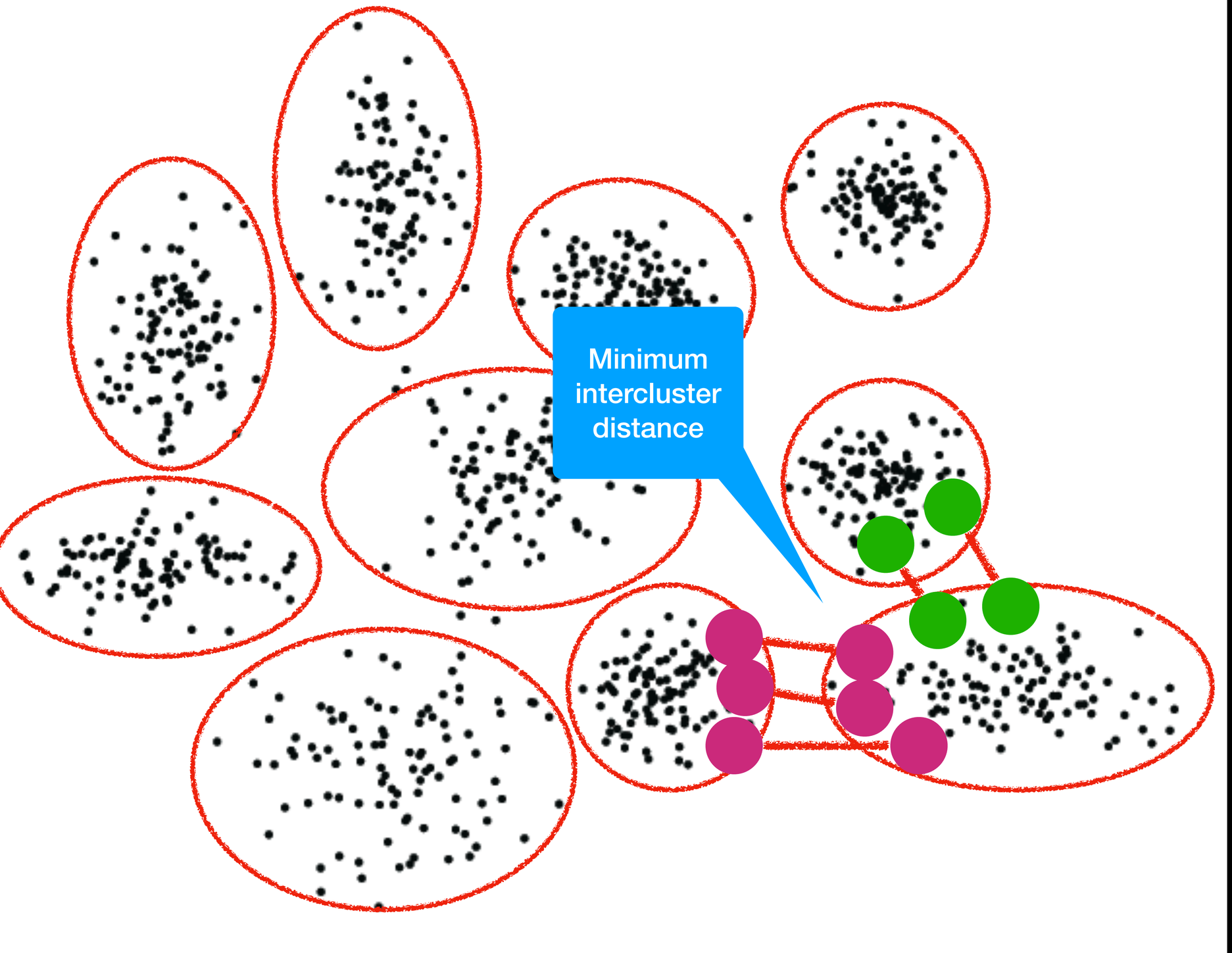
Explain how one can calculate distances between clusters

Determining “nearness” of two clusters

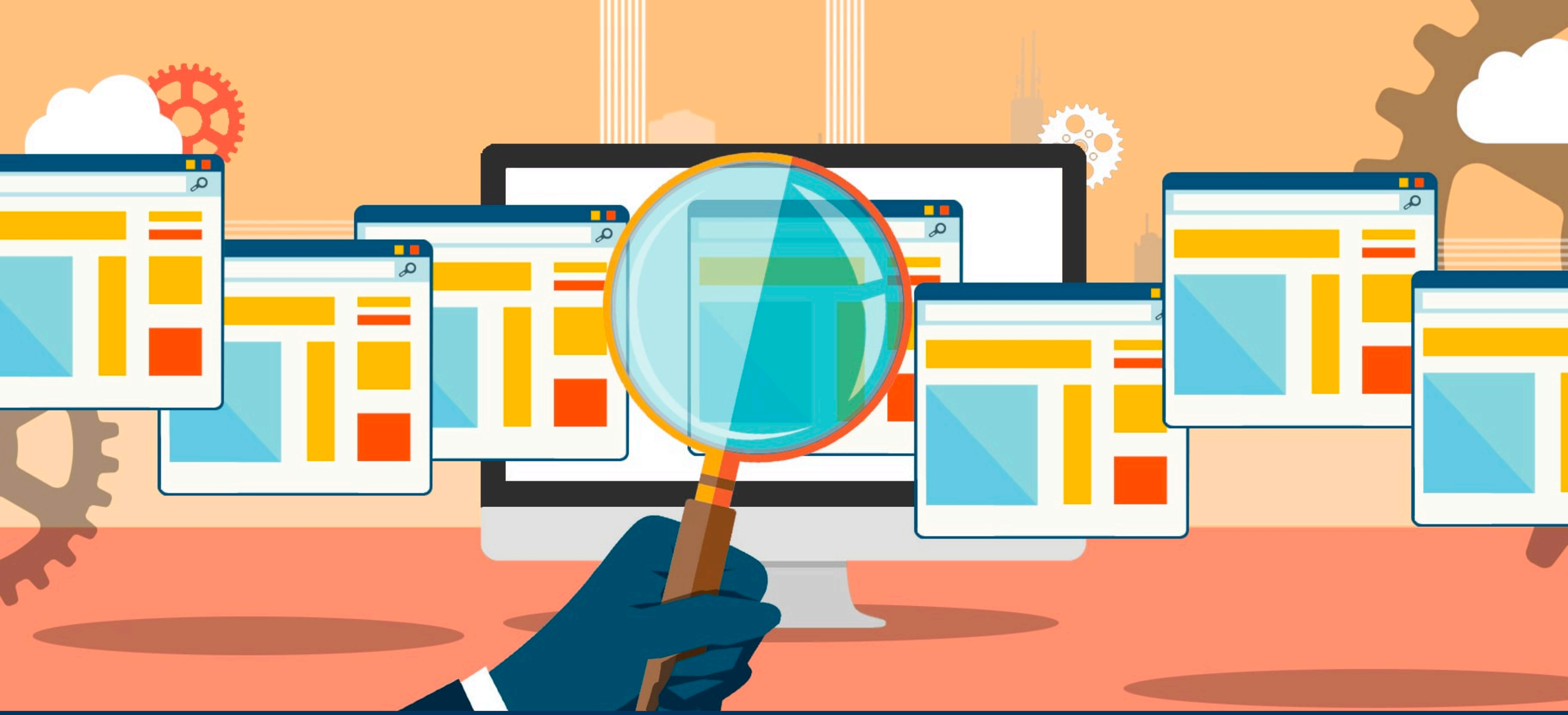
How do we measure the distance between two clusters?




Distance
between
centroids


Minimum
intercluster
distance

How might we actually perform clustering?



k-Means Clustering

INST414 - Data Science Techniques

This Module's Learning Objectives

Week 2

Explain how the k-means clustering algorithm identifies clusters

Describe at least two strategies for initializing clusters in k-means

Describe at least two strategies for selecting the number of clusters

This Module's Learning Objectives

Week 2

- Explain how the k-means clustering algorithm identifies clusters
- Describe at least two strategies for initializing clusters in k-means
- Describe at least two strategies for selecting the number of clusters

The k-mean Algorithm

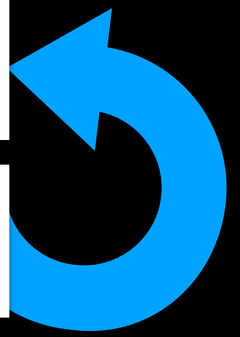
Step 1. Assess Euclidean distances between points

Step 2. Set your desired number of clusters ***k***

Step 3. Initialize a starting set of ***k*** clusters

Step 4. Assign all points to one of these ***k*** clusters

Step 5. Update your ***k*** clusters



Step 4. Assign all points to one of these ***k*** clusters

Step 4.1. For each point, identify the closest centroid

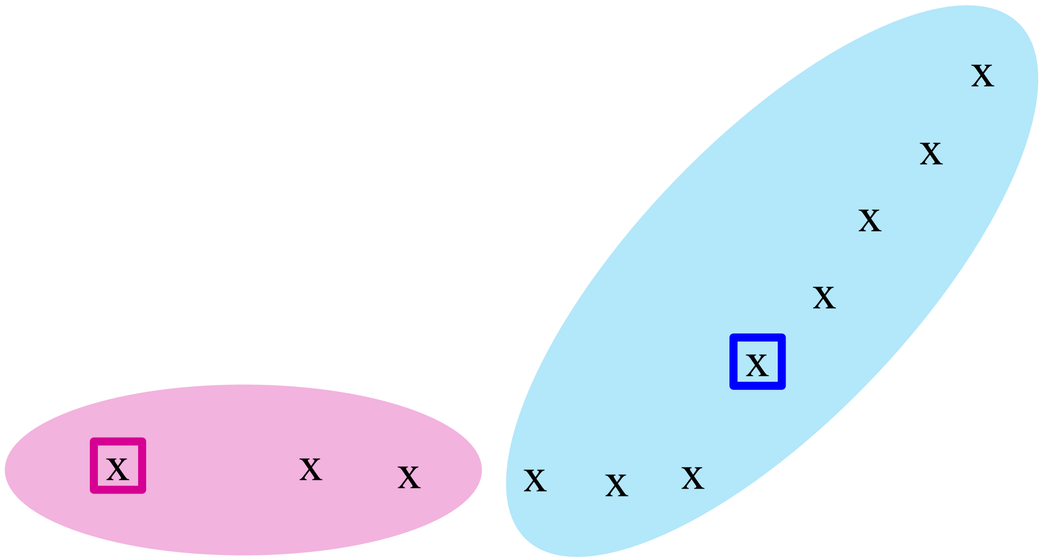
Step 4.2. Assign each point to the closest cluster ce

Step 5. Update your ***k*** clusters

Step 5.1. For each cluster, recalculate the centroid

Points may change clusters

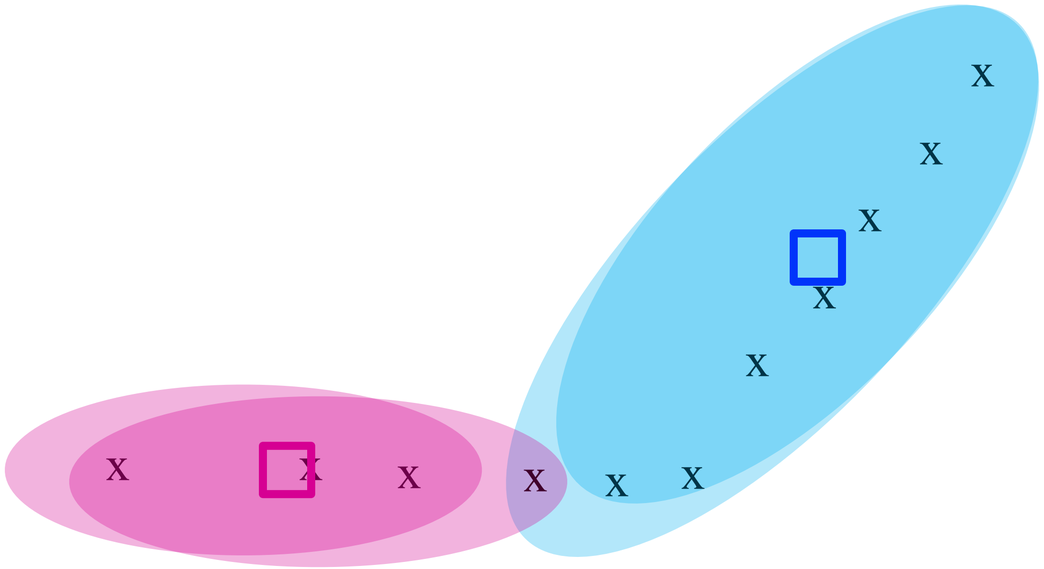
Example: Assigning Clusters



x … data point
□ … centroid

Clusters after round 1

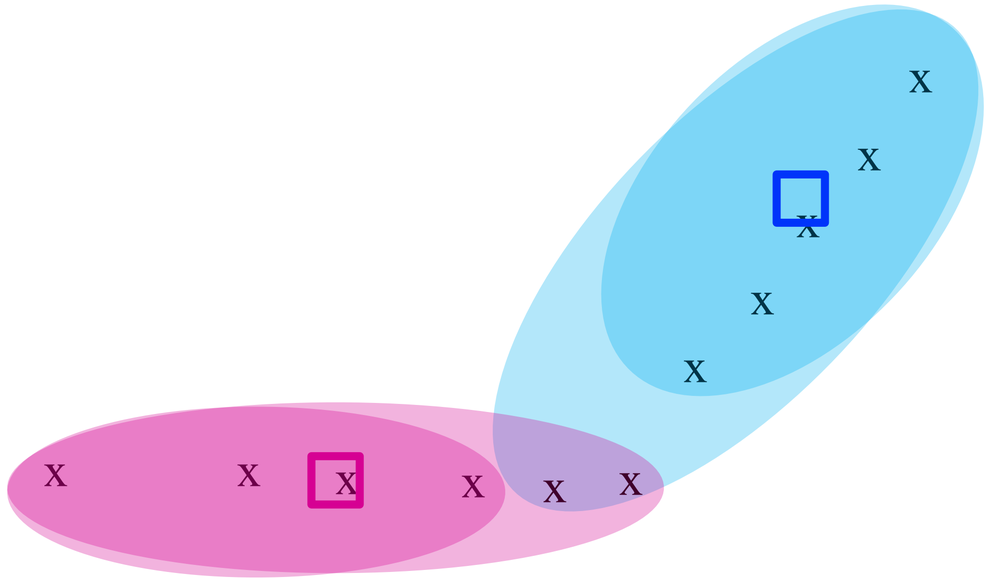
Example: Assigning Clusters



x … data point
□ … centroid

Clusters after round 2

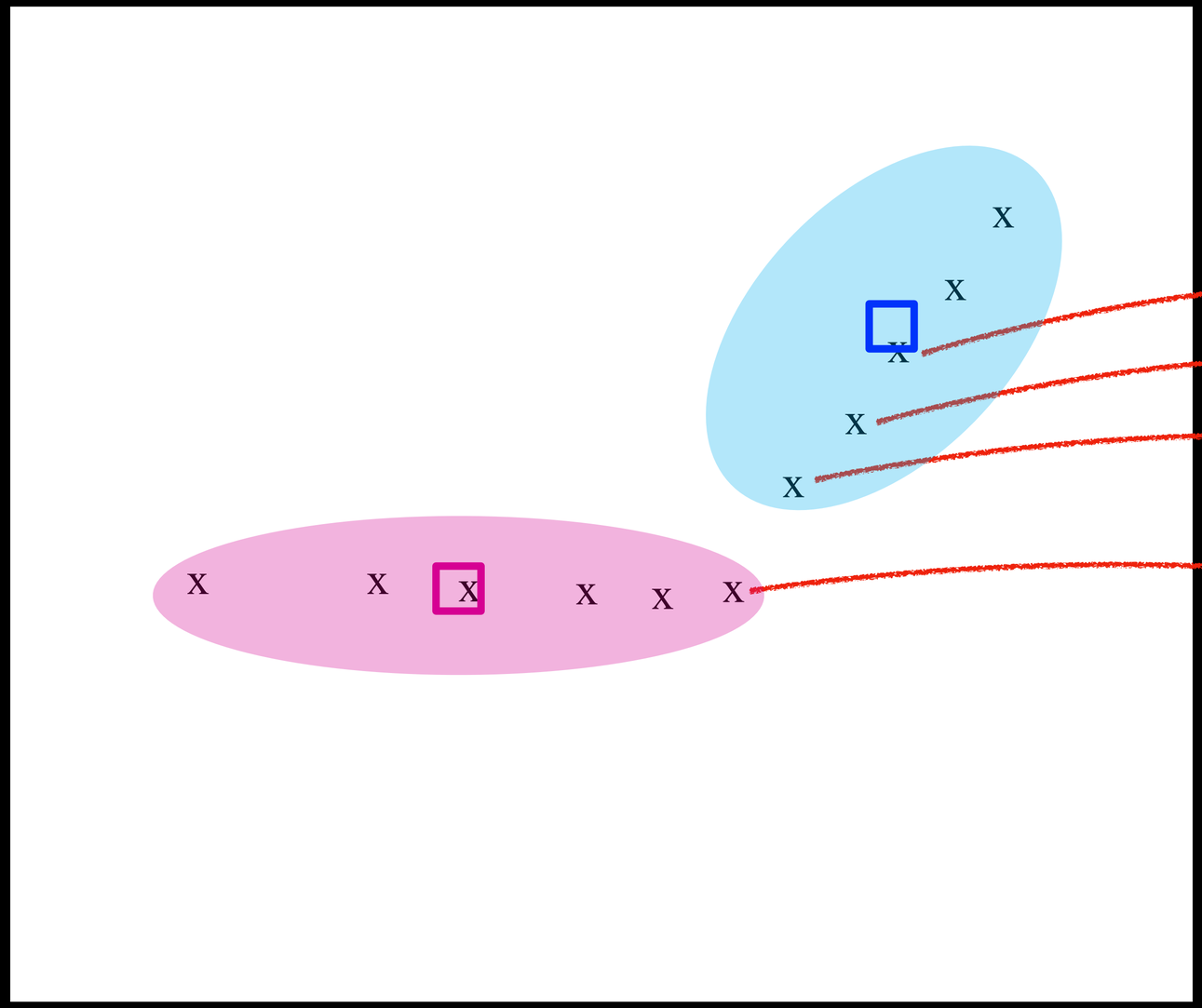
Example: Assigning Clusters



x … data point
□ … centroid

Clusters at the end

What's should clustering output be here?



Cluster Assignments

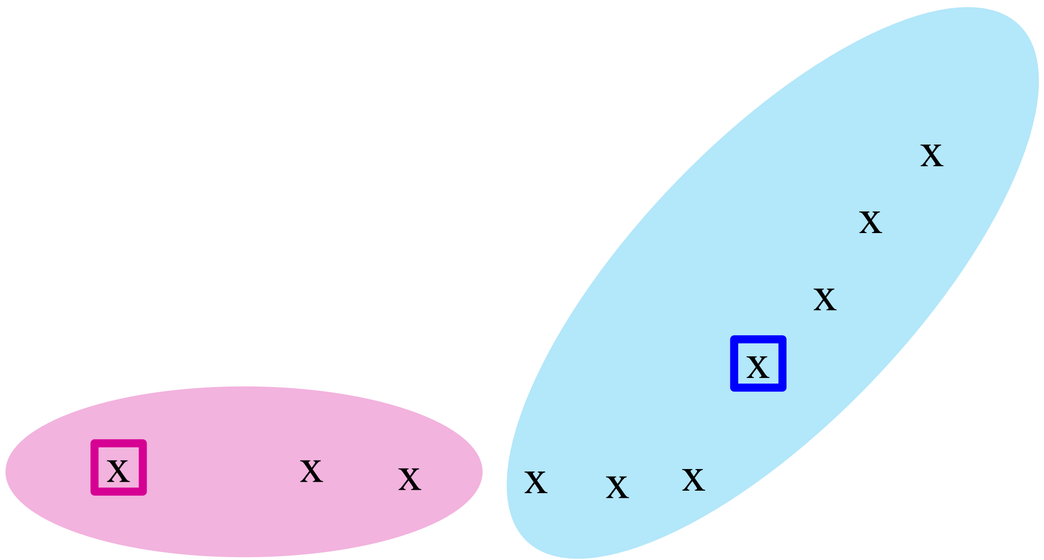
Point	Cluster
d1	0
d2	0
d3	0
…	…
dn	1

And we record centroid locations:

$C_0 = \langle x_0, y_0 \rangle$

$C_1 = \langle x_1, y_1 \rangle$

Why is it useful to record these cluster centers?



Were these good initial cluster centers?

What impact does this choice have?

The k-mean Algorithm

Step 1. Assess Euclidean distances between points

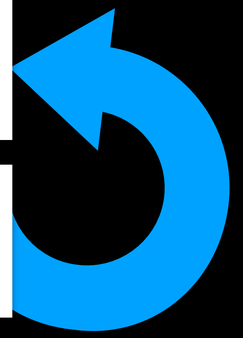
Step 2. Set your desired number of clusters ***k***

Step 3. Initialize a starting set of ***k*** clusters

Step 4. Assign all points to one of these ***k*** clusters

Step 5. Update your ***k*** clusters

Picking one point per cluster suggests we already know the clustering



How might we pick initial cluster centers?

What will happen if we pick poorly?

What will happen if we pick well?

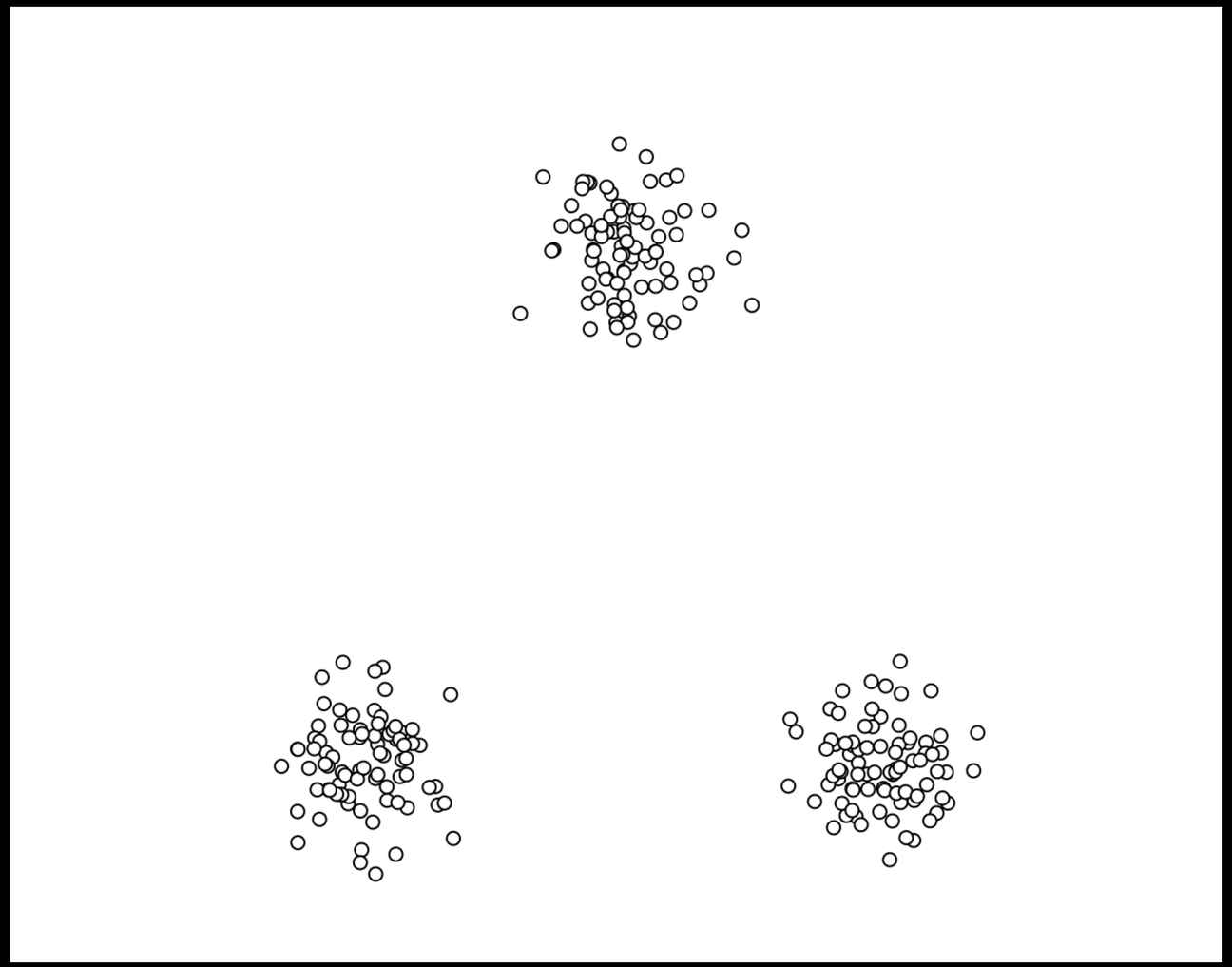
This Module's Learning Objectives

Week 2

Explain how the k-means clustering algorithm identifies clusters

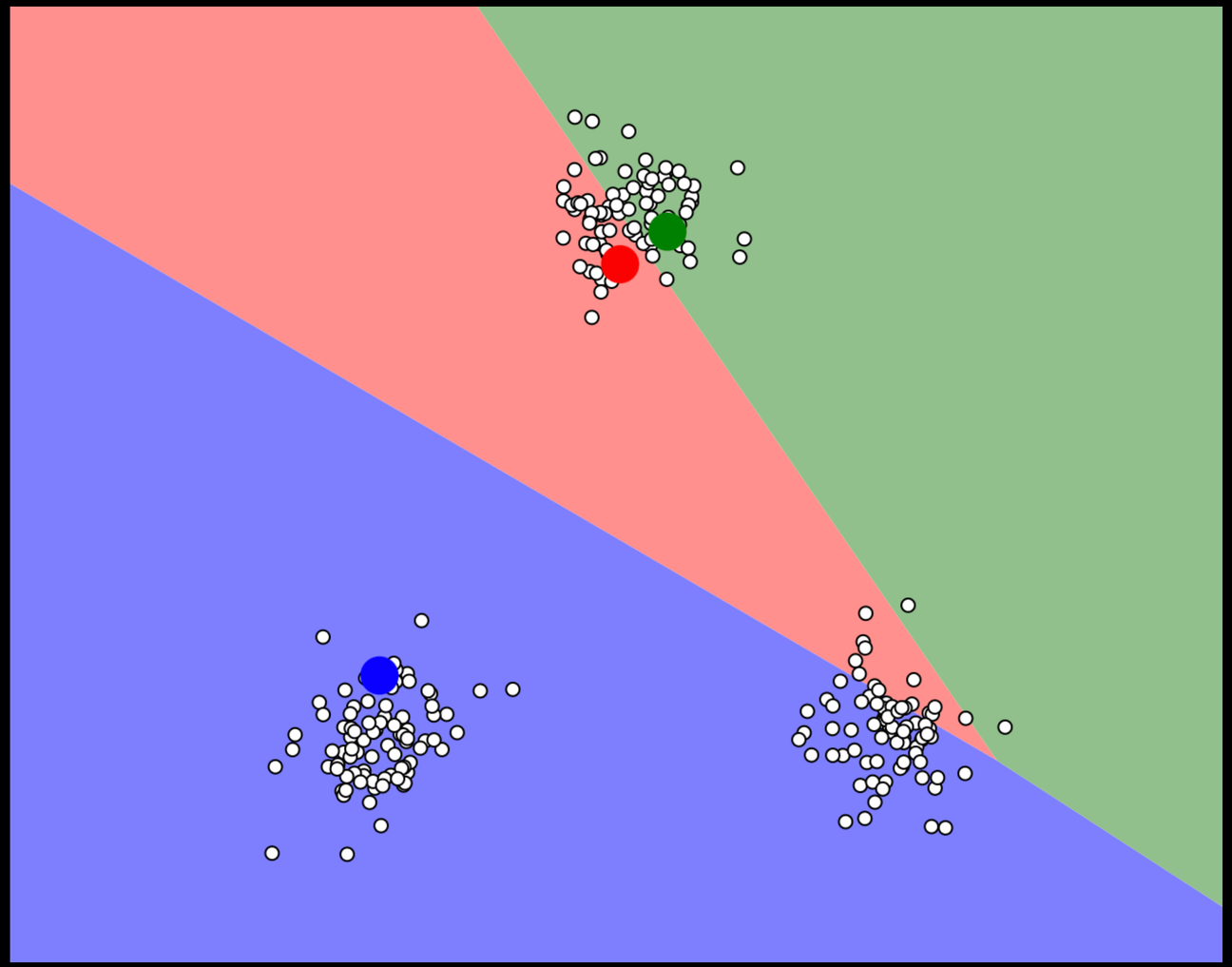
Describe at least two strategies for initializing clusters in k-means

Describe at least two strategies for selecting the number of clusters



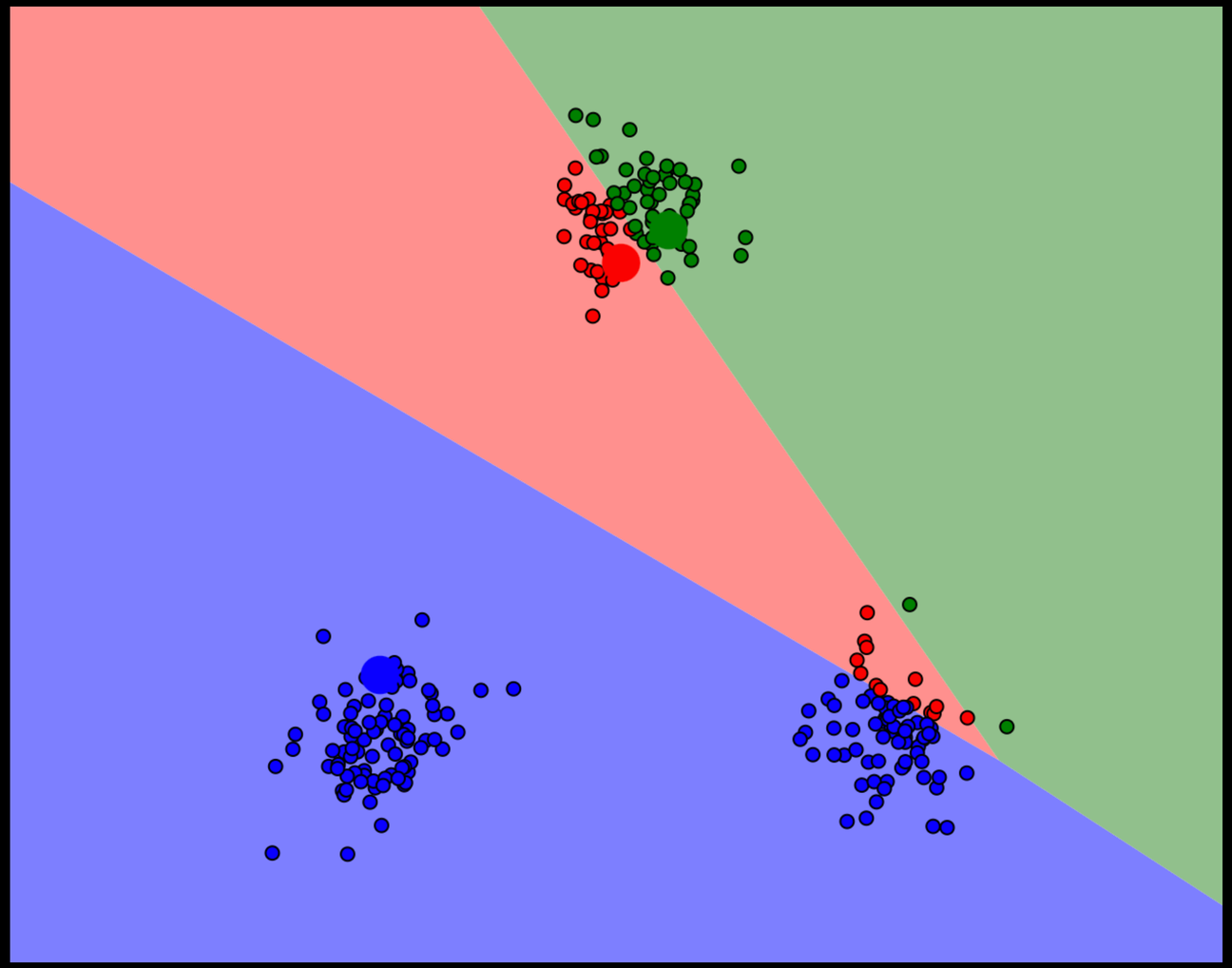
Naive solution: **Pick *k* points at random from data**

Likely to hit modes in the data



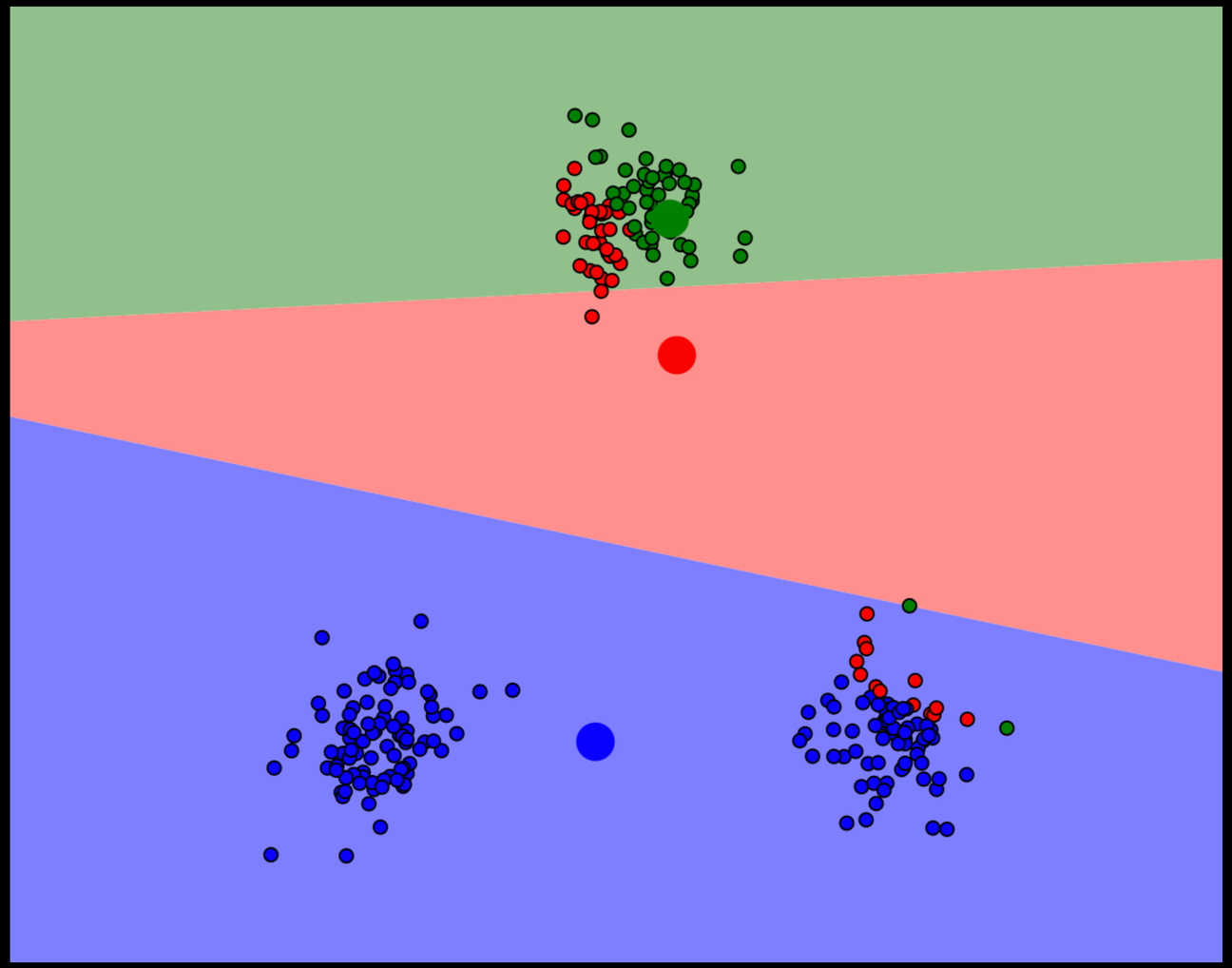
Naive solution: **Pick *k* points at random from data**

Likely to hit modes in the data



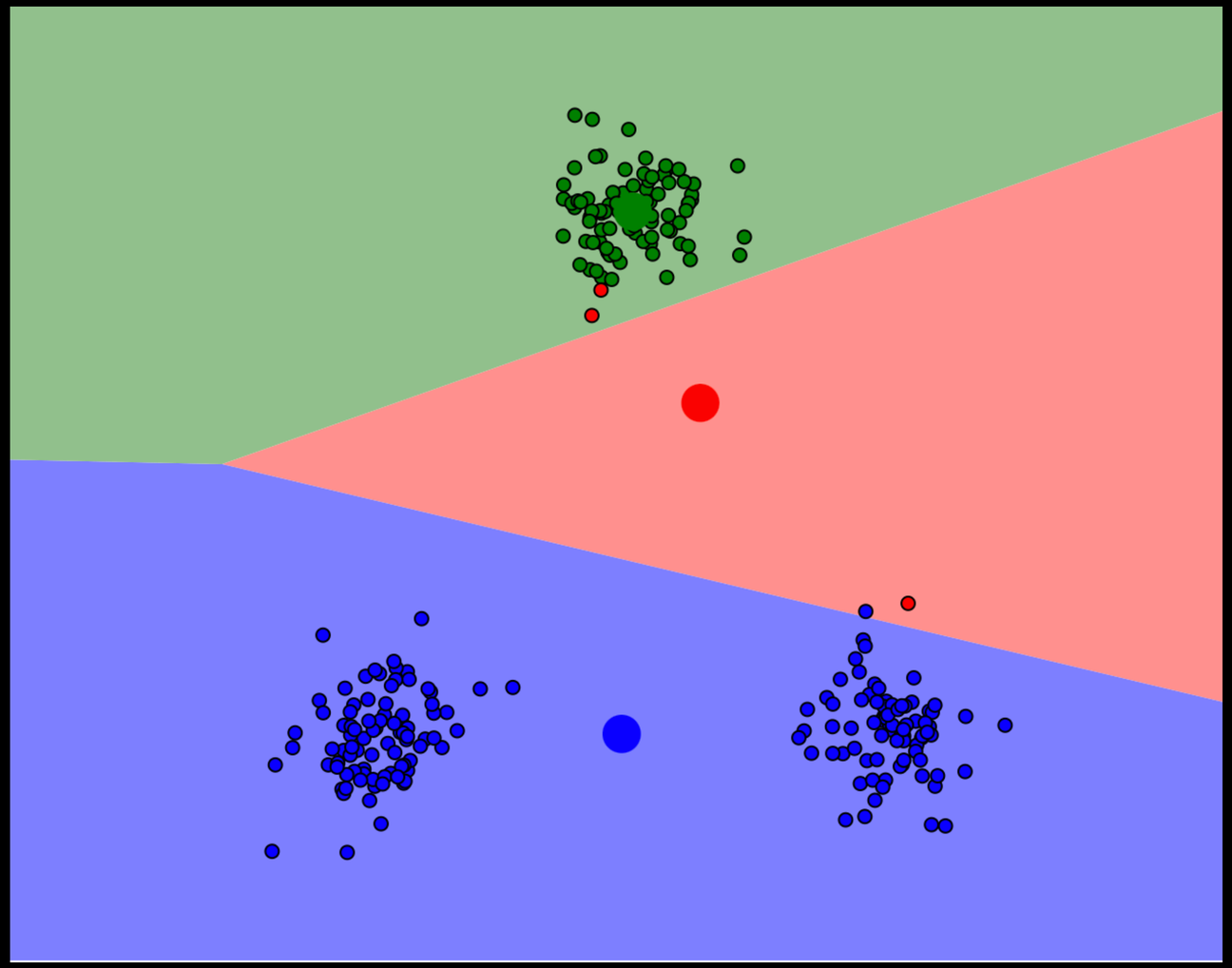
Naive solution: **Pick *k* points at random from data**

Likely to hit modes in the data



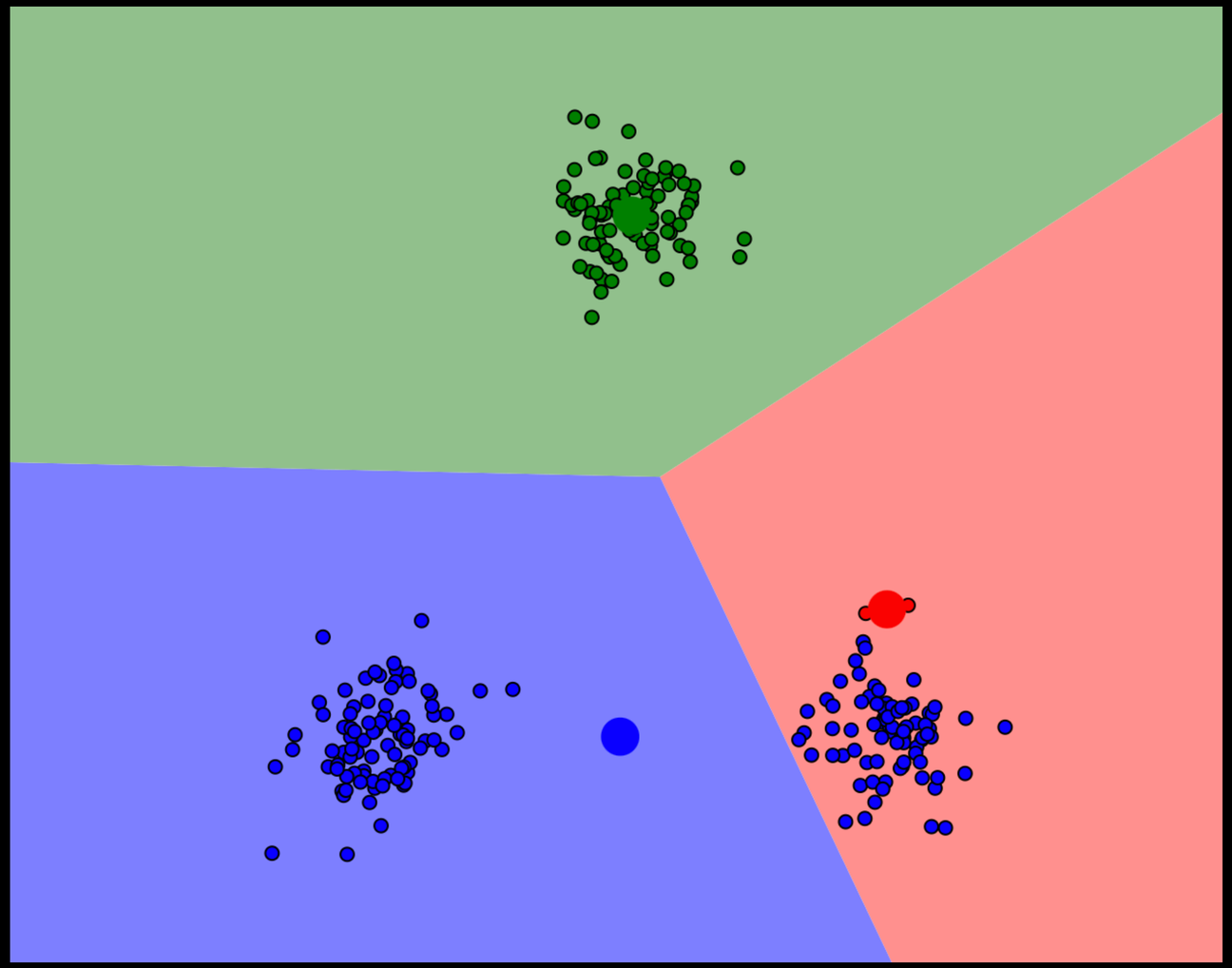
Naive solution: **Pick *k* points at random from data**

Likely to hit modes in the data



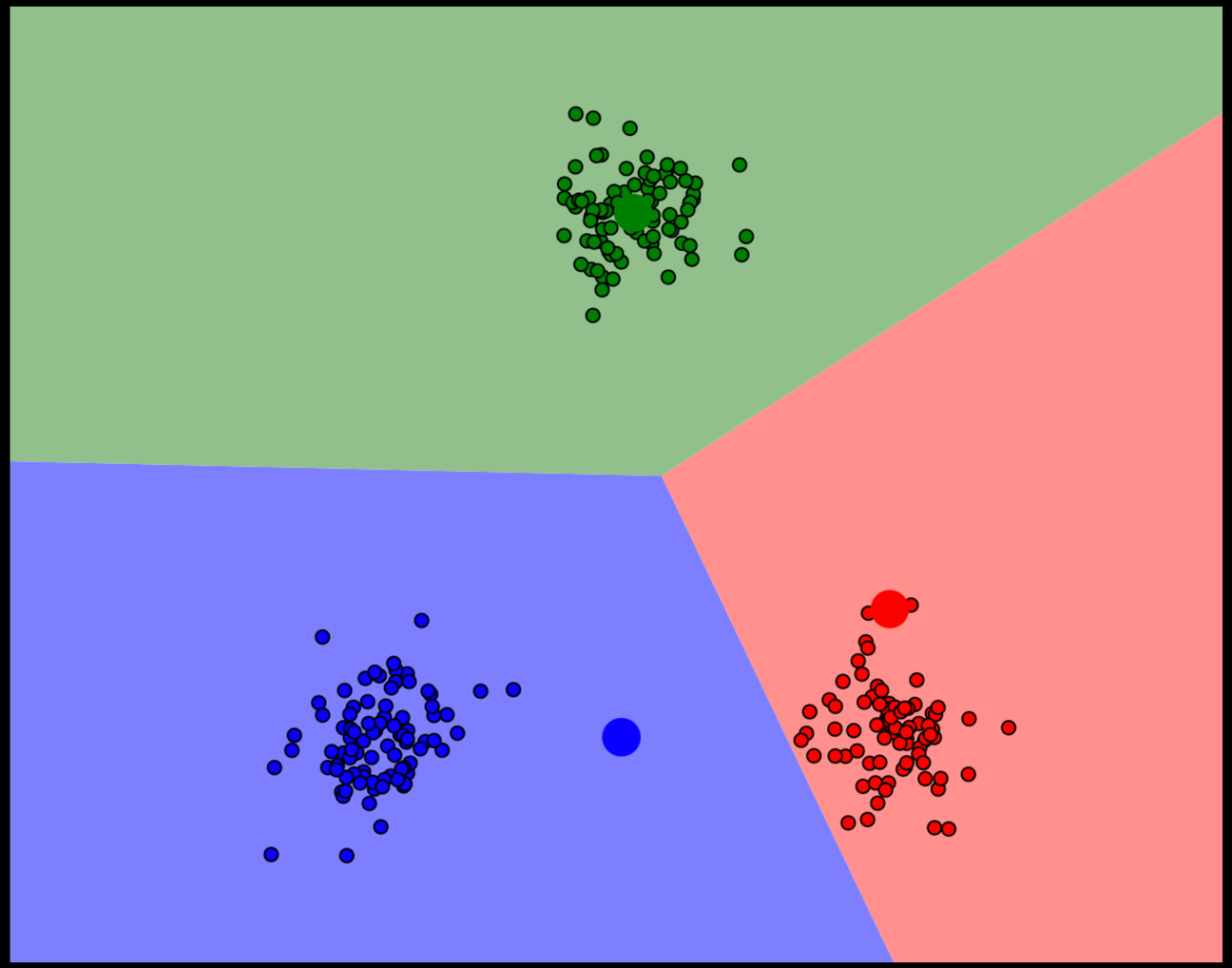
Naive solution: **Pick *k* points at random from data**

Likely to hit modes in the data



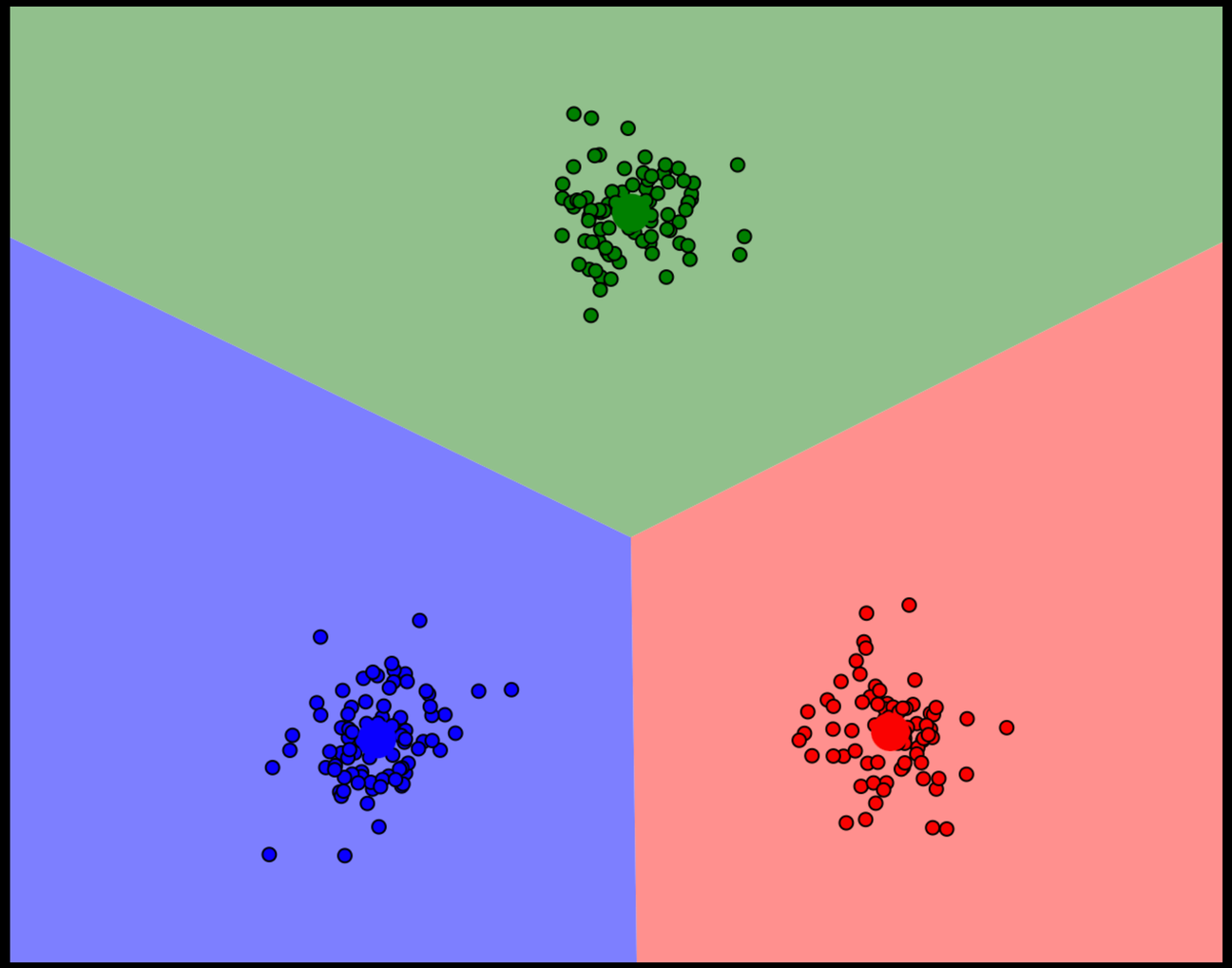
Naive solution: **Pick *k* points at random from data**

Likely to hit modes in the data



Naive solution: **Pick *k* points at random from data**

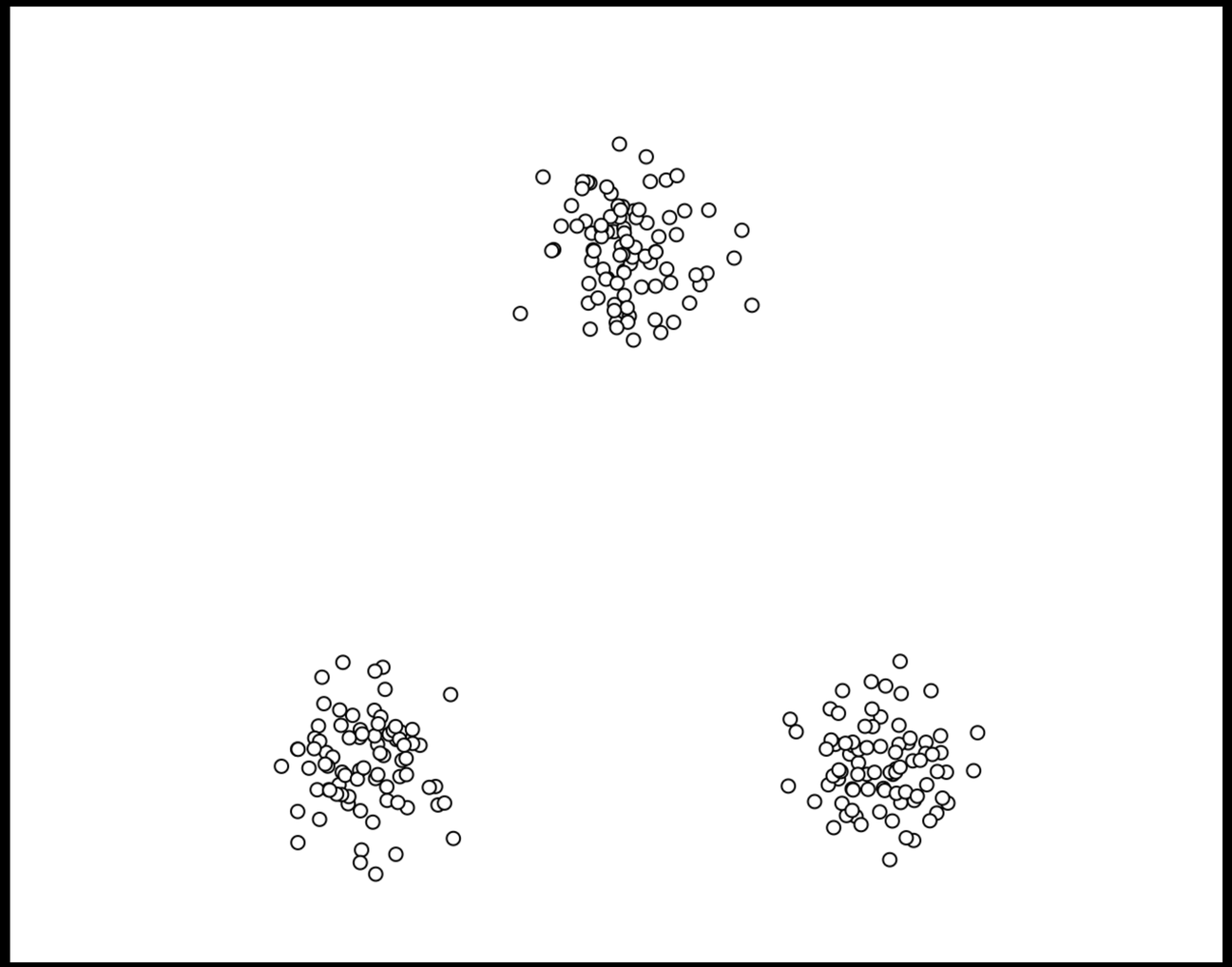
Likely to hit modes in the data



Naive solution: **Pick *k* points at random from data**

Likely to hit modes in the data

What happens if we pick bad initial cluster points?

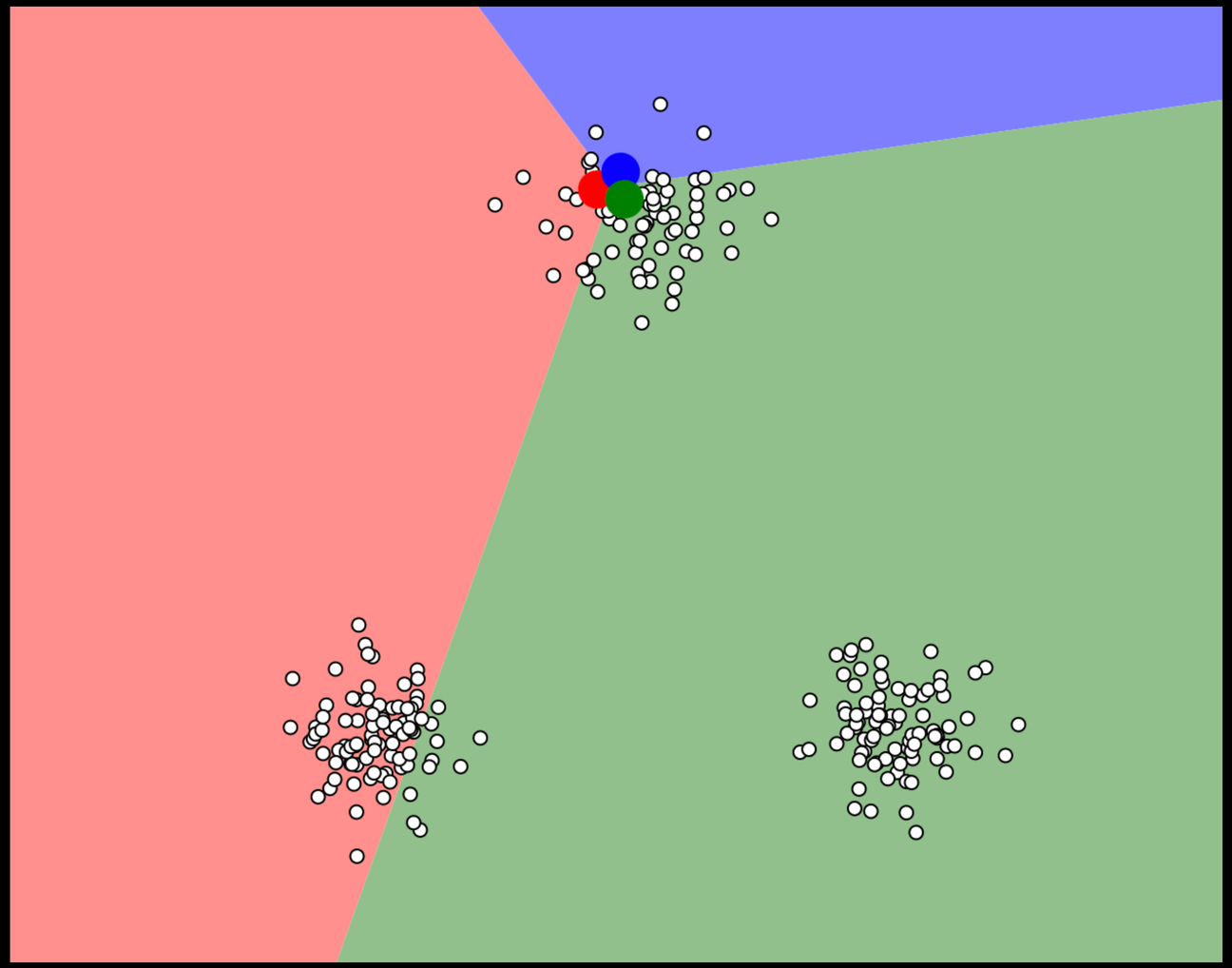


Naive solution: **Pick *k* points at random from data**

Likely to hit modes in the data

Implications of poor selection:

Slower convergence

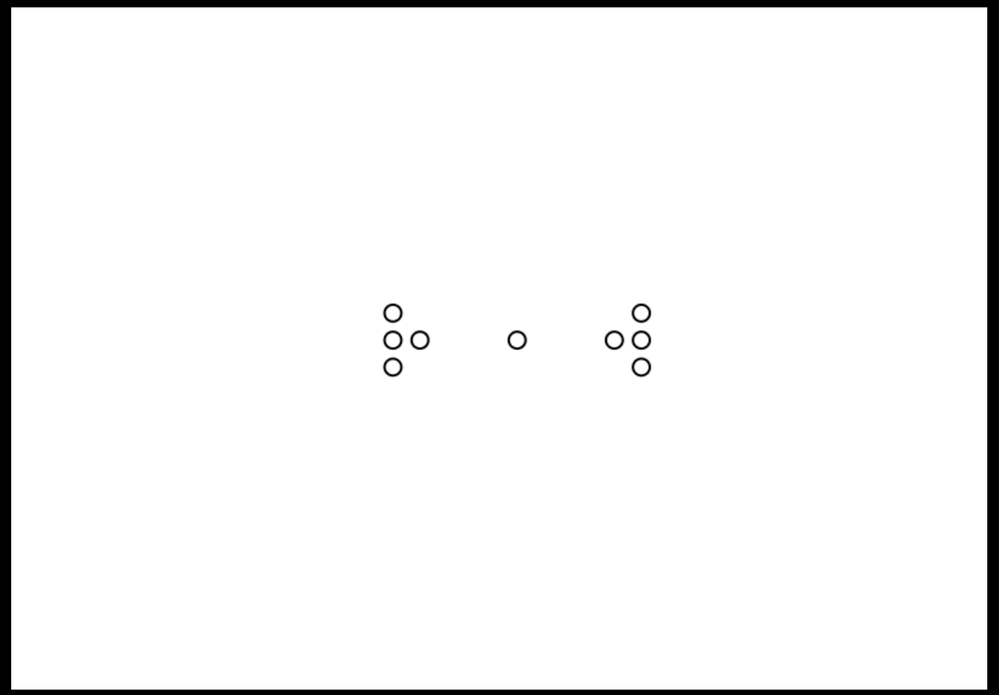


Naive solution: **Pick *k* points at random from data**

Likely to hit modes in the data

Implications of poor selection:

Slower convergence



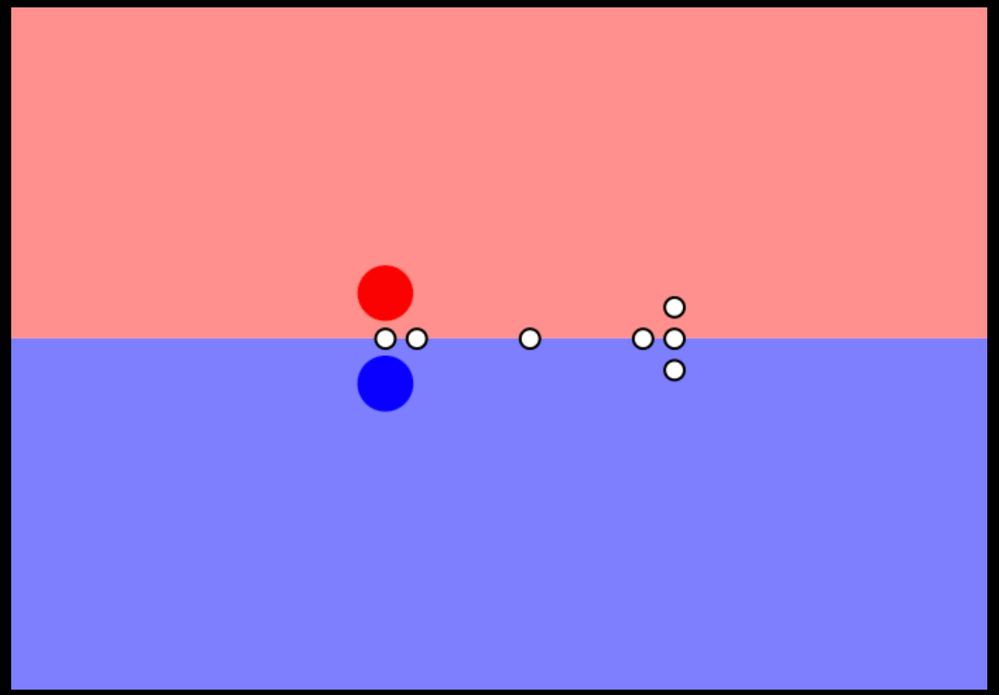
Naive solution: **Pick *k* points at random from data**

Likely to hit modes in the data

Implications of poor selection:

Slower convergence

Non-optimal final clustering



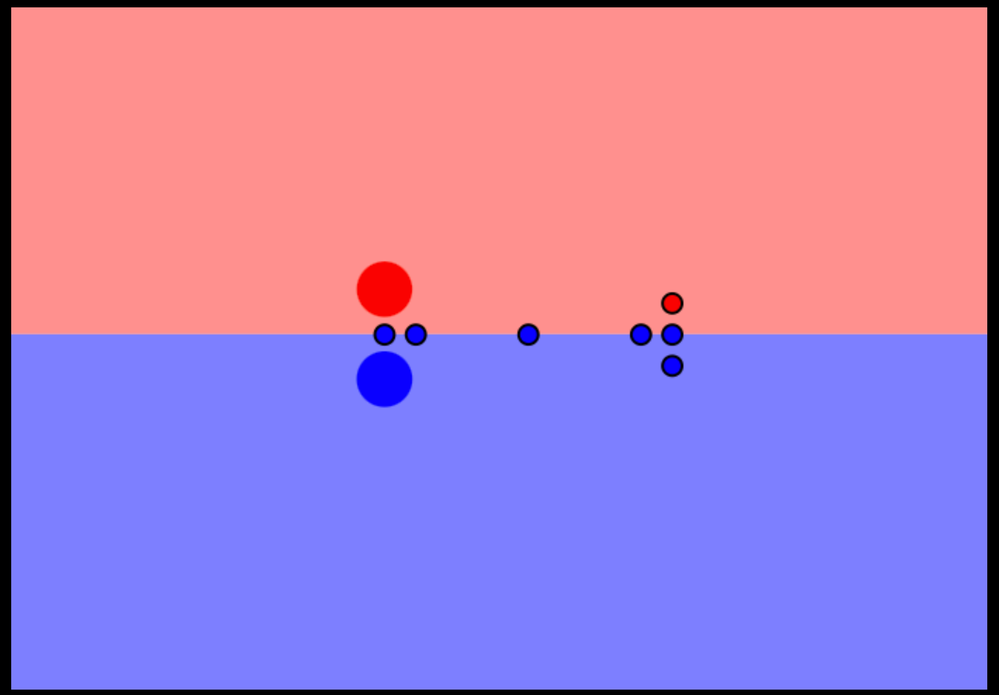
Naive solution: **Pick *k* points at random from data**

Likely to hit modes in the data

Implications of poor selection:

Slower convergence

Non-optimal final clustering



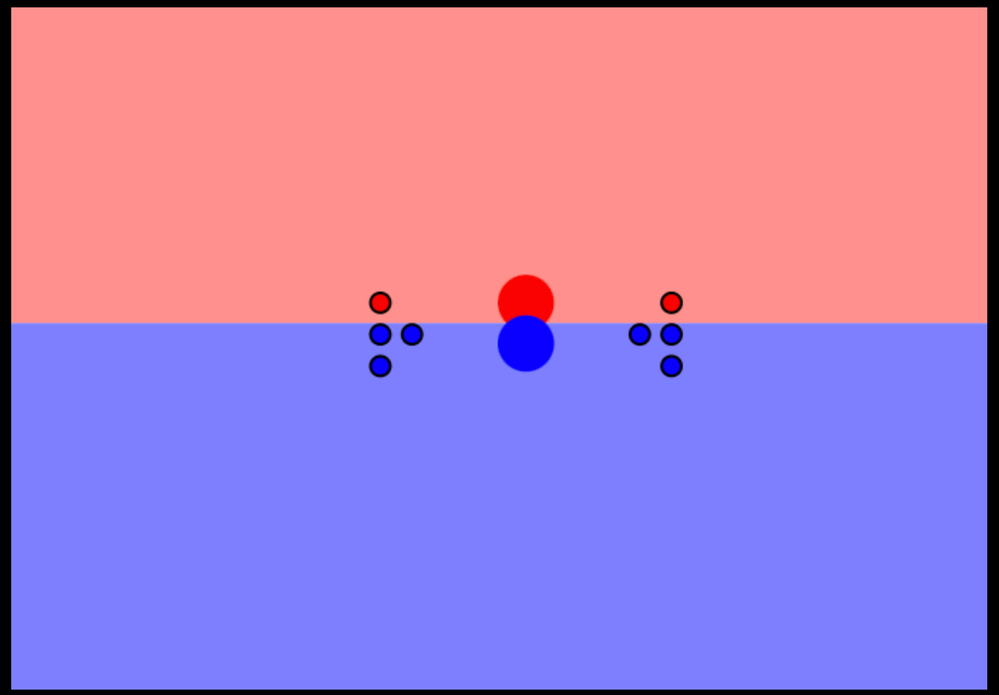
Naive solution: **Pick *k* points at random from data**

Likely to hit modes in the data

Implications of poor selection:

Slower convergence

Non-optimal final clustering



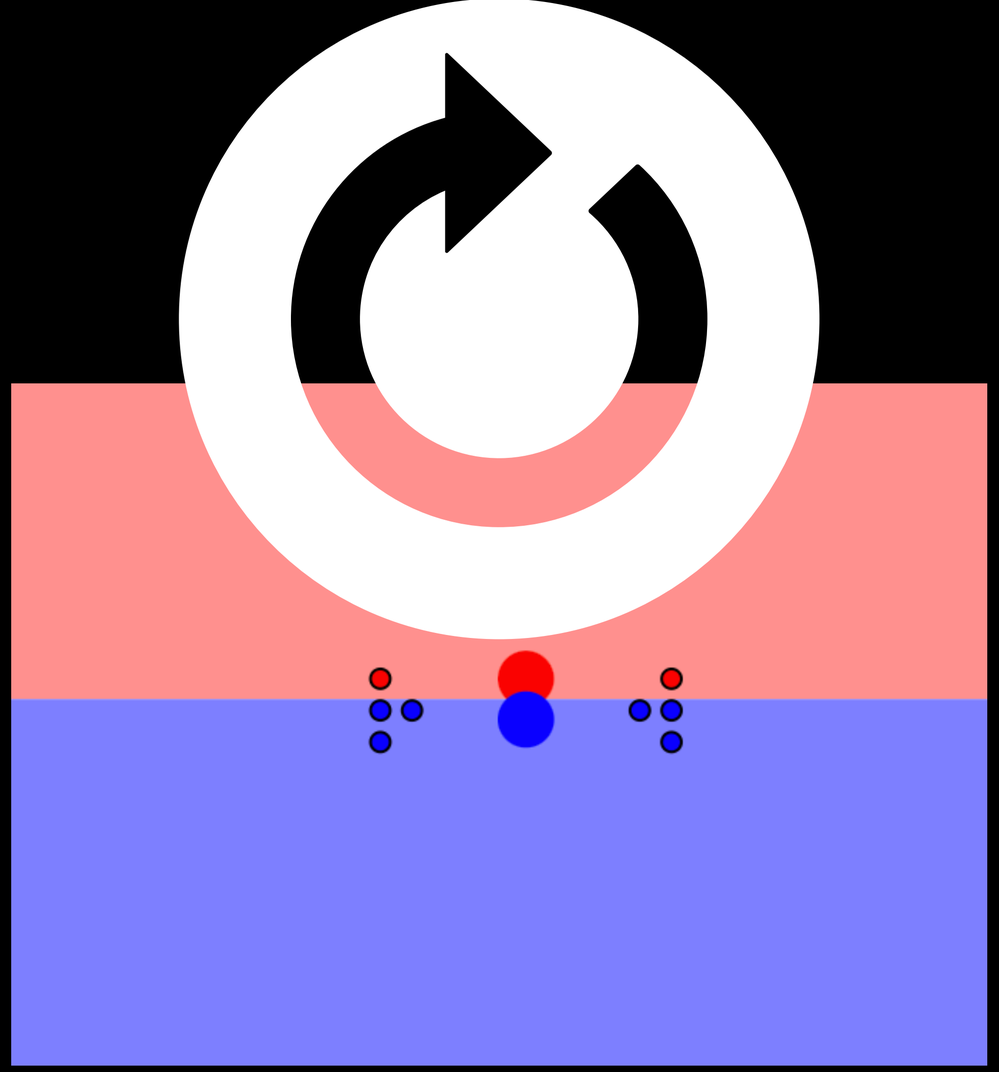
Naive solution: **Pick *k* points at random from data**

Likely to hit modes in the data

Implications of poor selection:

Slower convergence

Non-optimal final clustering



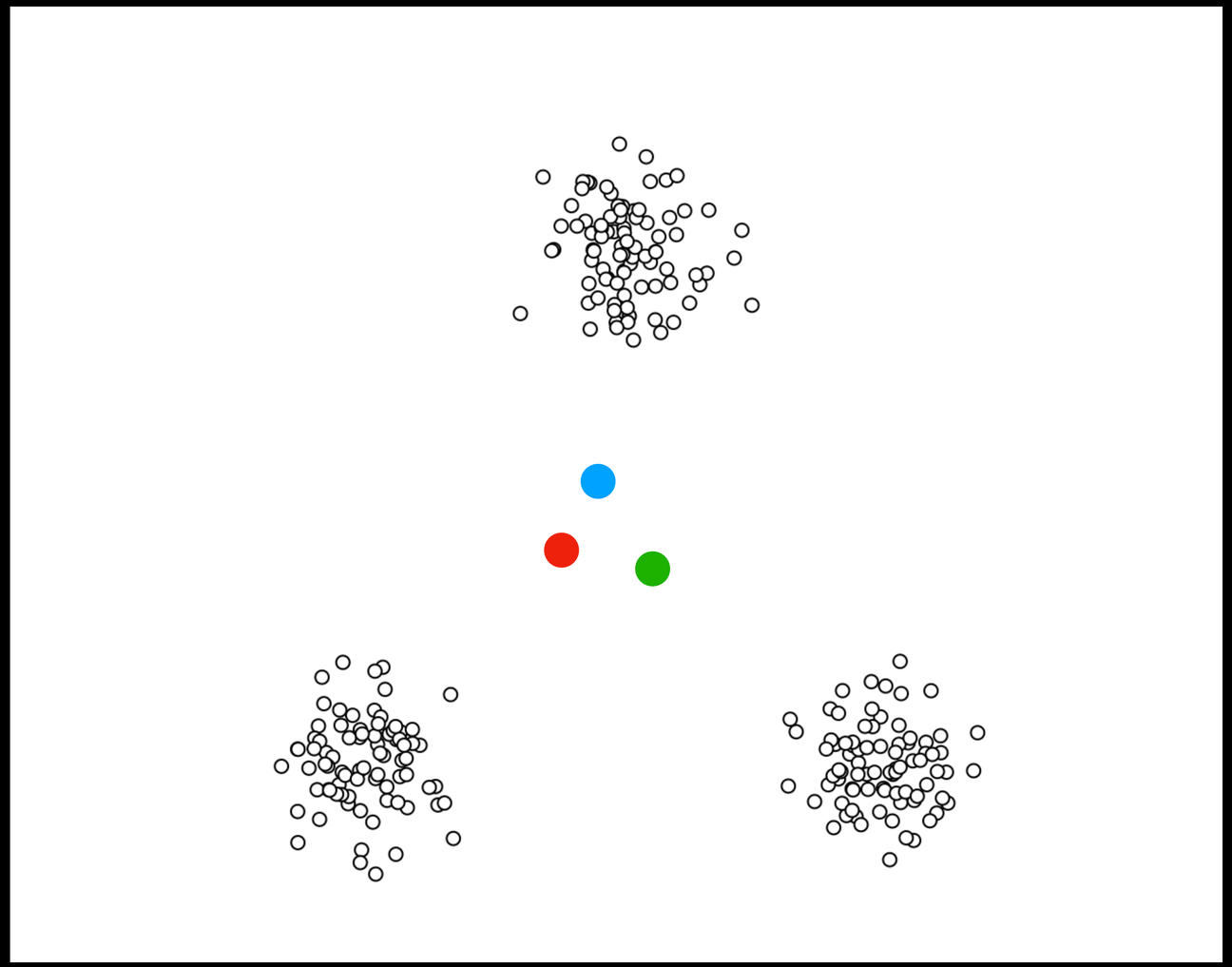
Naive solution: **Pick *k* points at random from data**

Likely to hit modes in the data

Implications of poor selection:

Slower convergence

Non-optimal final clustering



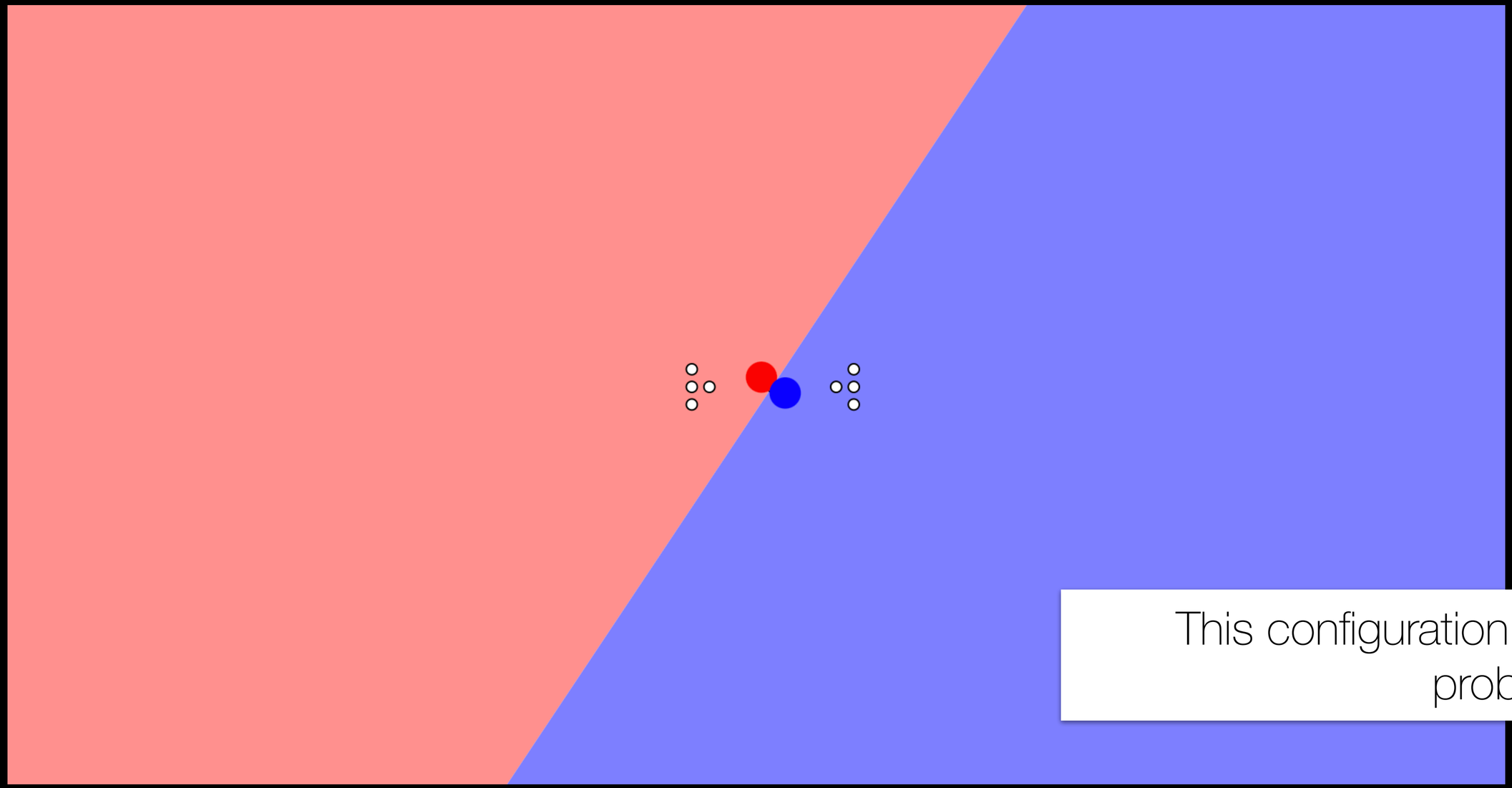
Alternative: Randomly partition data into ***k*** sets, and init w/ centroids

Use centers of these random clusters

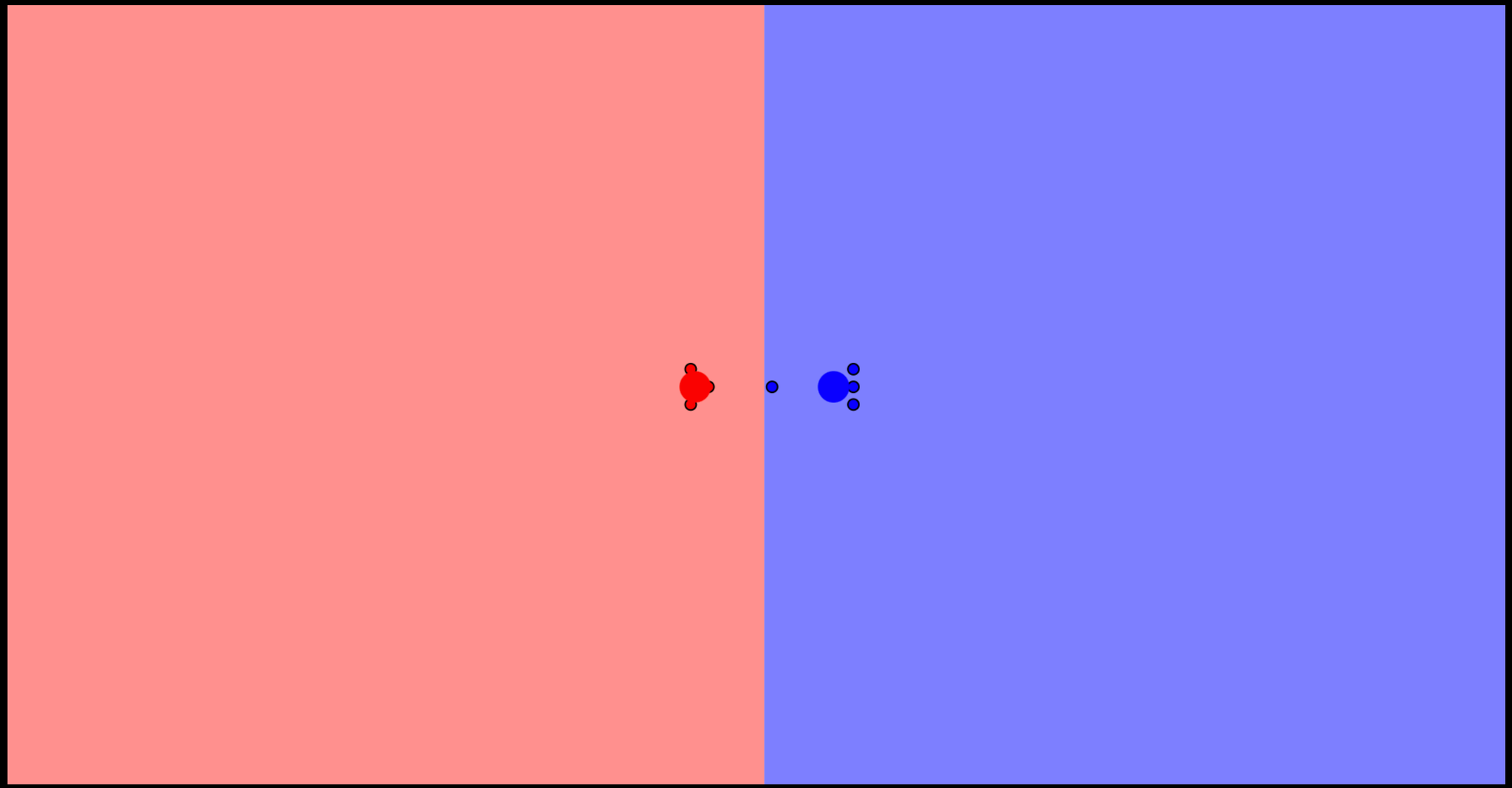
Problematic solution…

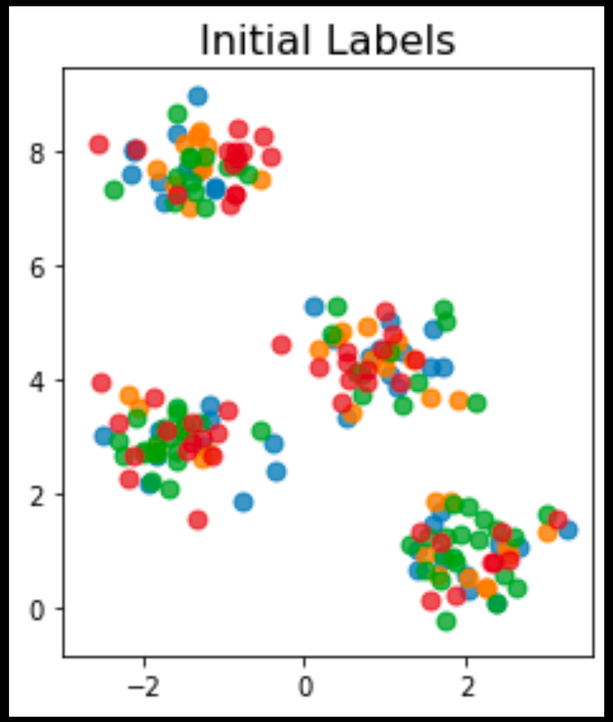
What is a random partition's mean?

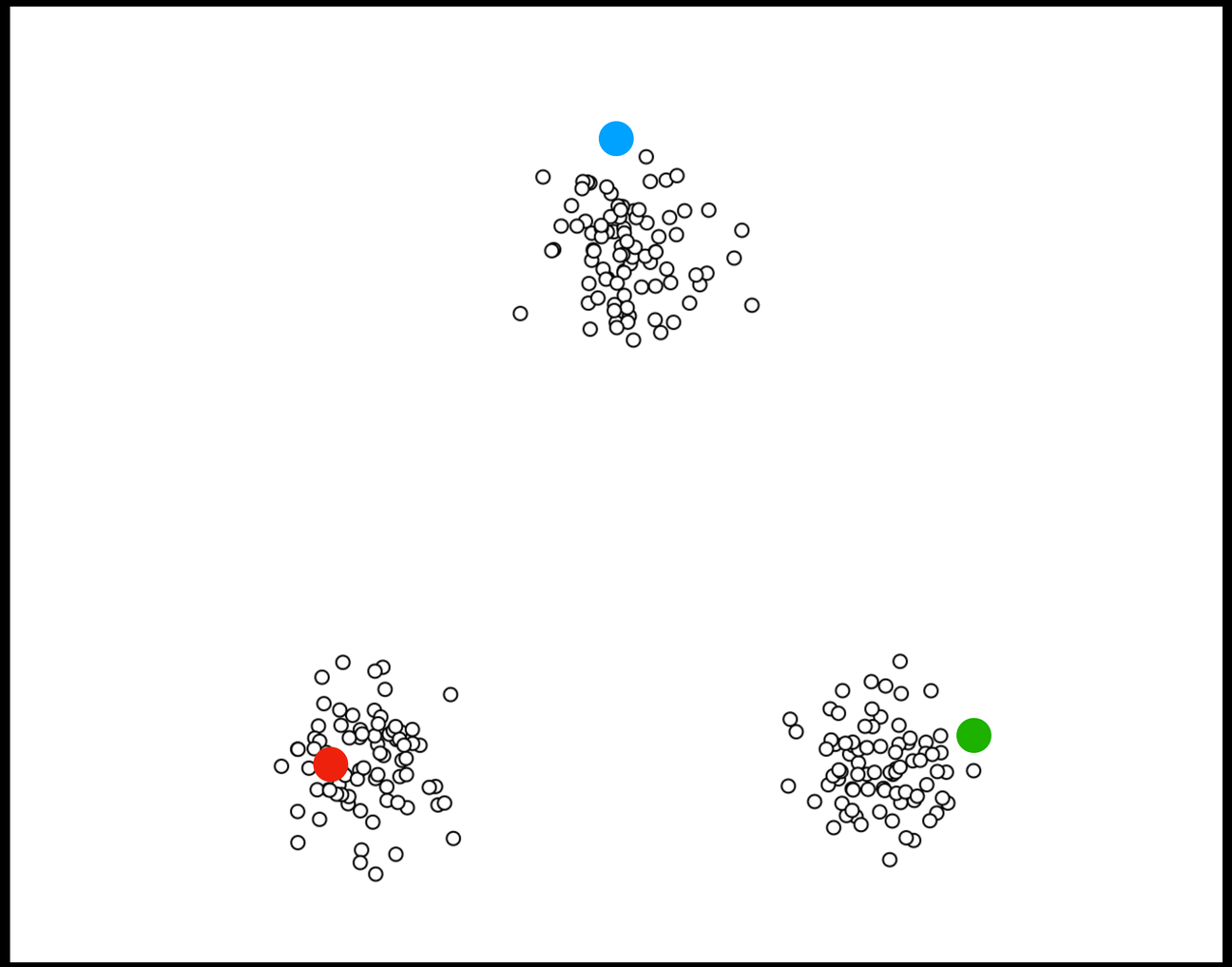
Likely near the global mean



This configuration can be problematic




Initial Labels
8
6
4
2
0
−2
0
2



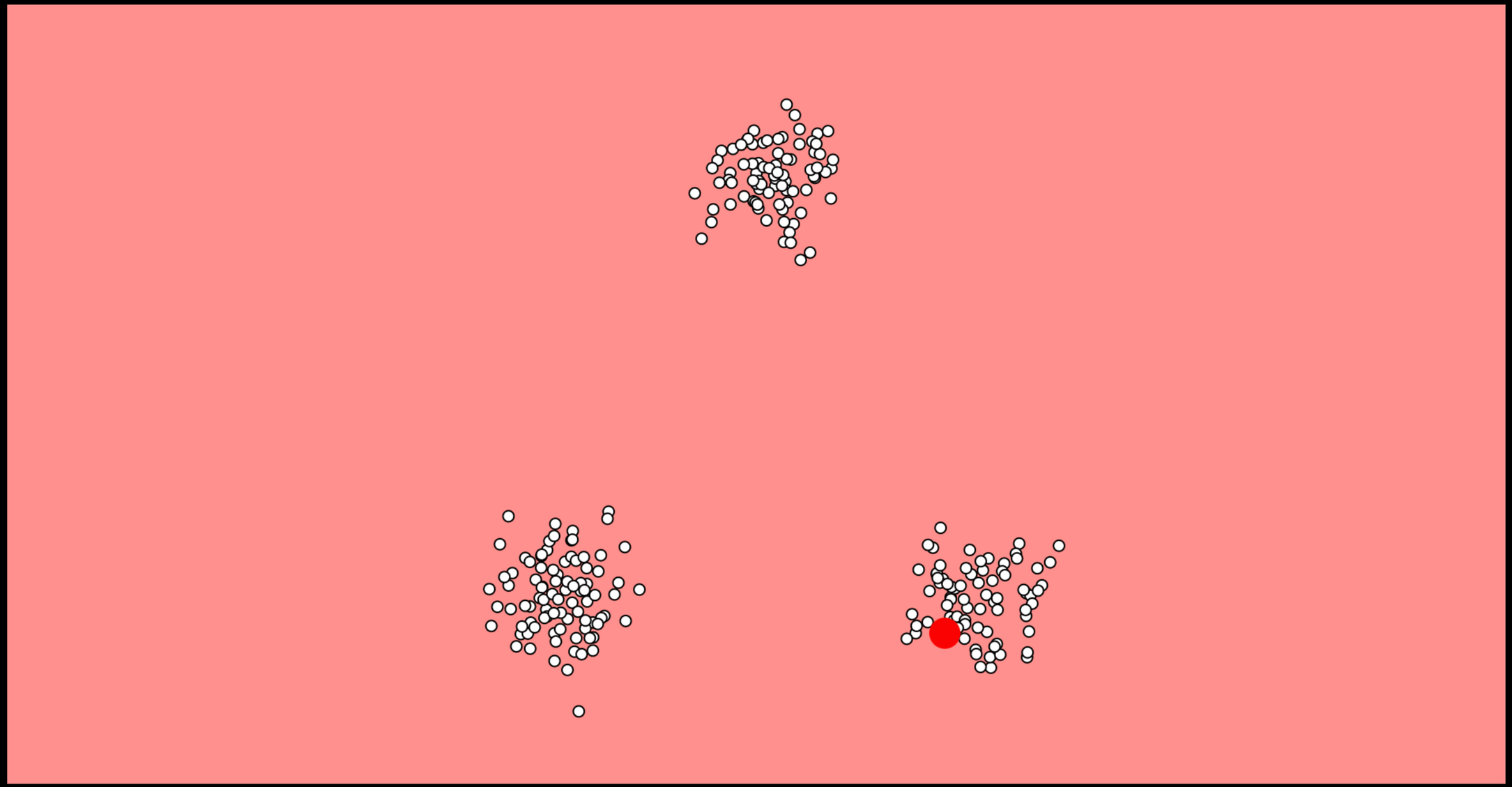
Better Solution: Pick ***k*** points that are far from each other

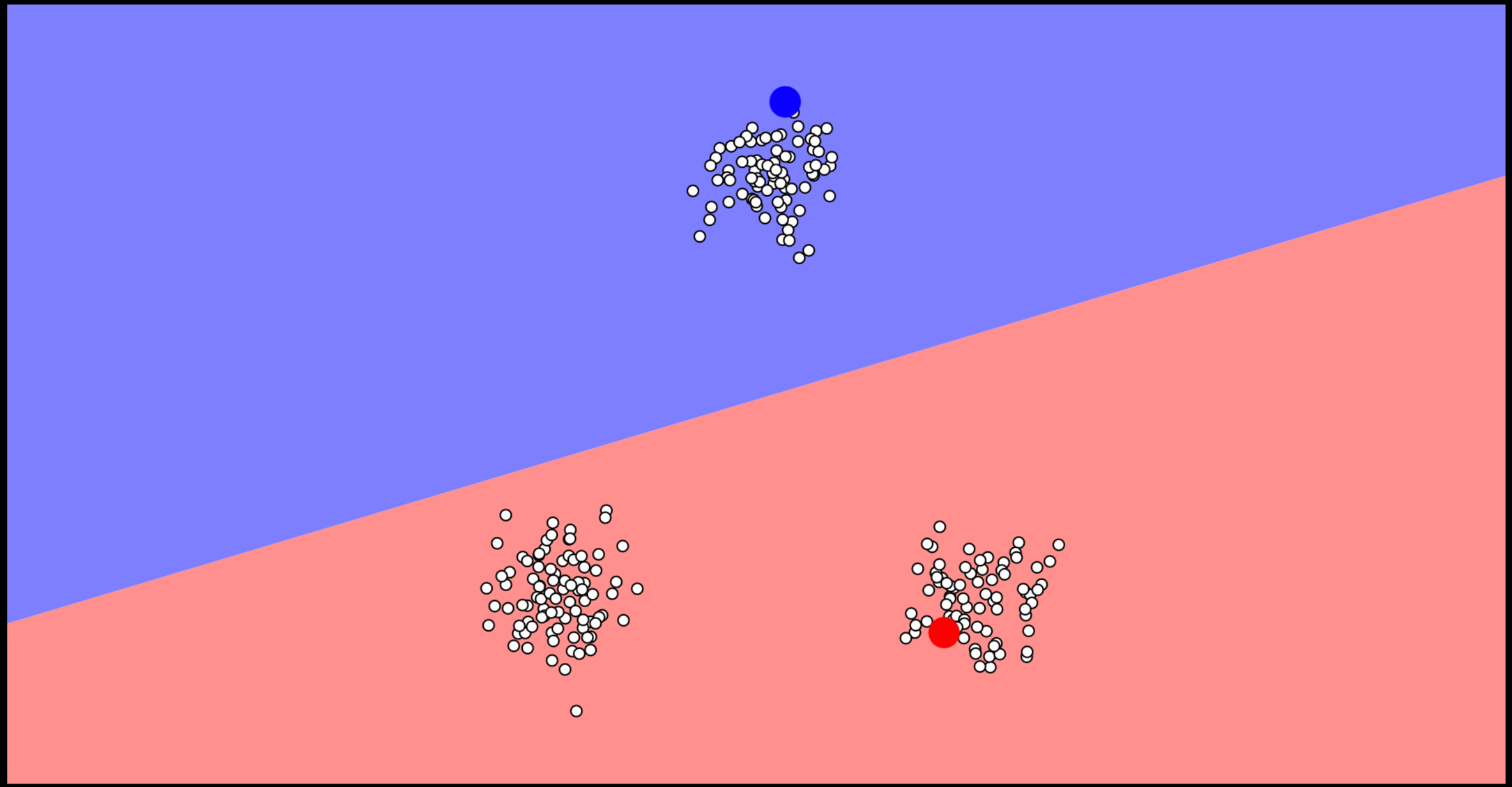
Intuition: Cluster centers should be far from each other

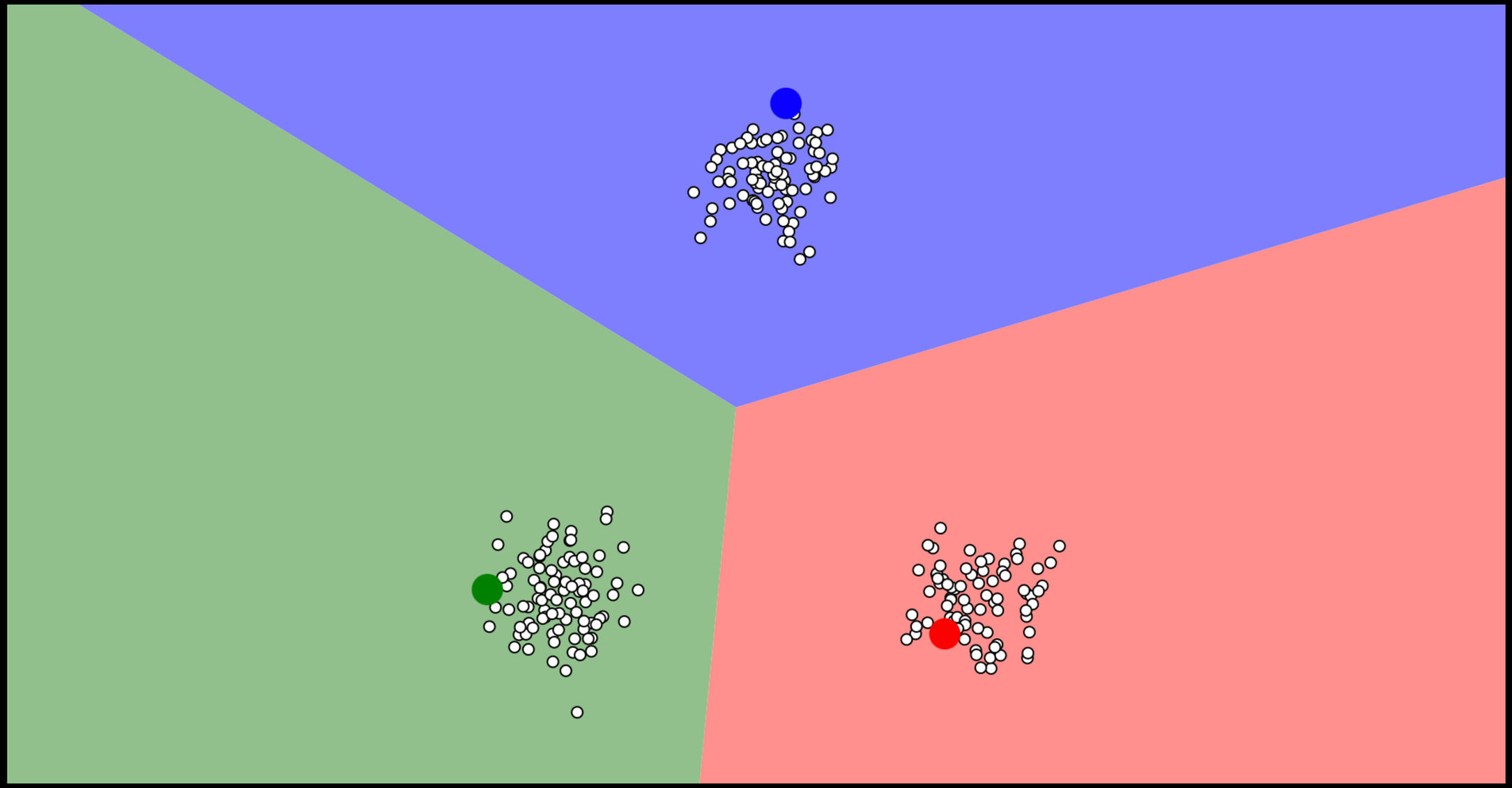
Premise of k-Means++

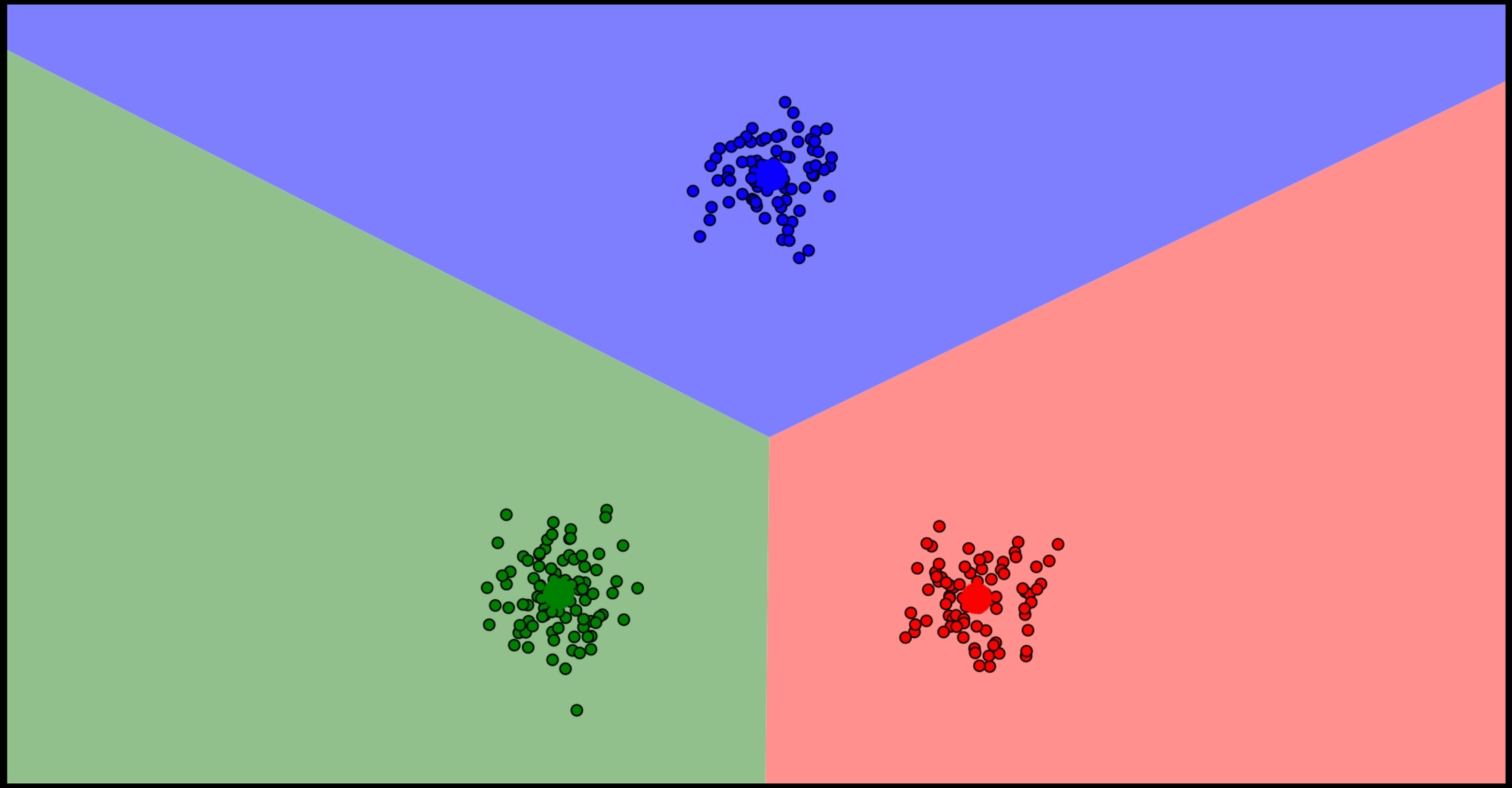
Pick a random point

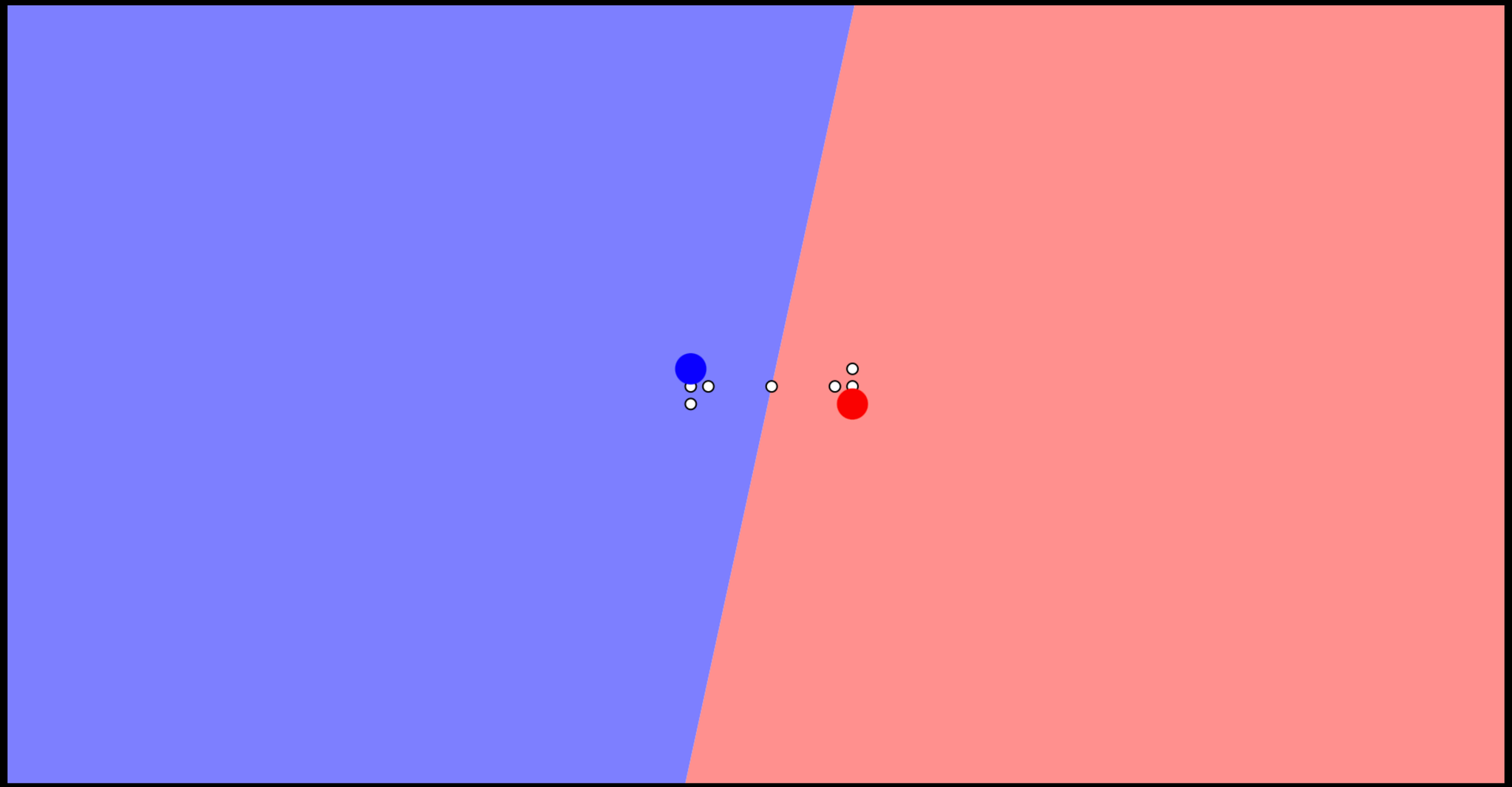
Iteratively pick next point so it's far away

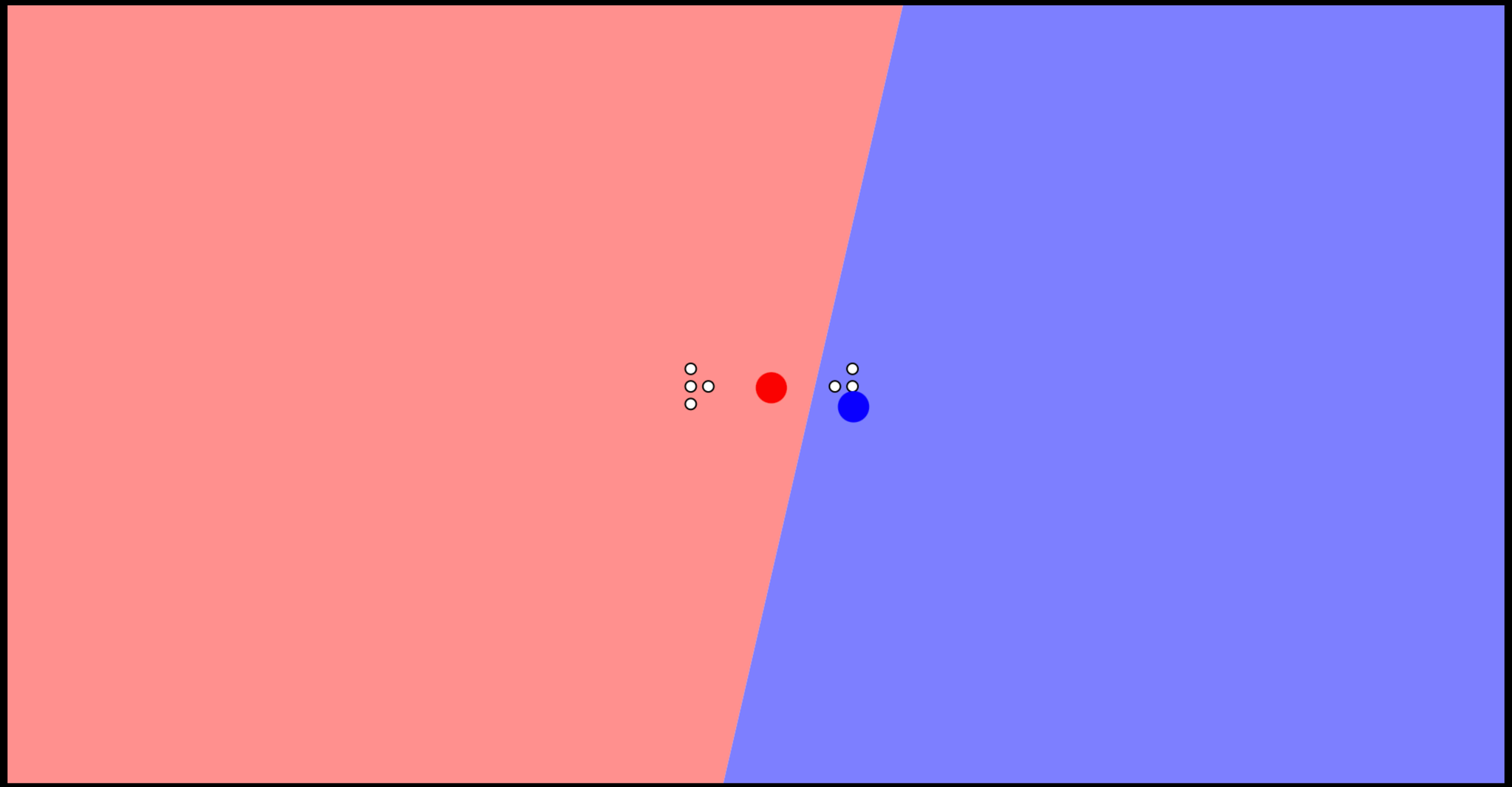


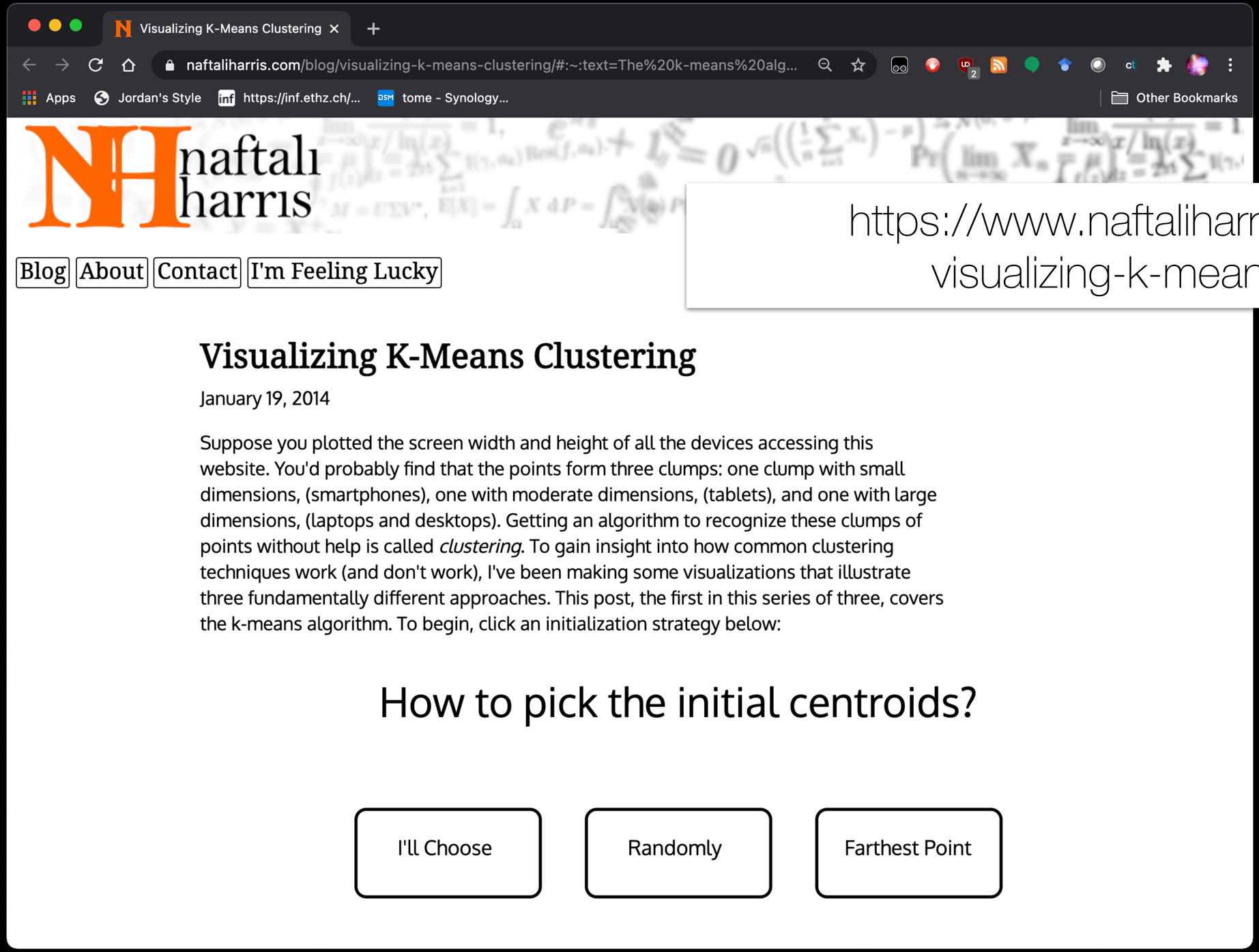











Visualizing K-Means Clustering
naftaliharris.com/blog/visualizing-k-means-clustering/#:~:text=The%20k-means%20alg...
Apps
Jordan's Style
https://inf.ethz.ch/...
tome - Synology...
Other Bookmarks
naftali harris
Blog
About
Contact
I'm Feeling Lucky
https://www.naftaliharris.com/blog/visualizing-k-means-clustering/
Visualizing K-Means Clustering
January 19, 2014
Suppose you plotted the screen width and height of all the devices accessing this website. You'd probably find that the points form three clumps: one clump with small dimensions, (smartphones), one with moderate dimensions, (tablets), and one with large dimensions, (laptops and desktops). Getting an algorithm to recognize these clumps of points without help is called clustering. To gain insight into how common clustering techniques work (and don't work), I've been making some visualizations that illustrate three fundamentally different approaches. This post, the first in this series of three, covers the k-means algorithm. To begin, click an initialization strategy below:
How to pick the initial centroids?
I'll Choose
Randomly
Farthest Point

Expert Systems with Applications 40 (2013) 200–210

Contents lists available at SciVerse ScienceDirect

Expert Systems with Applications

journal homepage: www.elsevier.com/locate/eswa

A comparative study of efficient initialization methods for the k-means clustering algorithm

M. Emre Celebi [a,*], Hassan A. Kingravi [b], Patricio A. Vela [b]

[a] Department of Computer Science, Louisiana State University, Shreveport, LA, USA
[b] School of Electrical and Computer Engineering, Georgia Institute of Technology, Atlanta, GA, USA

ARTICLE INFO

Keywords:
Partitional clustering
Sum of squared error criterion
k-means
Cluster center initialization

ABSTRACT

K-means is undoubtedly the most widely used partitional clustering algorithm. Unfortunately, due to its gradient descent nature, this algorithm is highly sensitive to the initial placement of the cluster centers. Numerous initialization methods have been proposed to address this problem. In this paper, we first present an overview of these methods with an emphasis on their computational efficiency. We then compare eight commonly used linear time complexity initialization methods on a large and diverse collection of data sets using various performance criteria. Finally, we analyze the experimental results using nonparametric statistical tests and provide recommendations for practitioners. We demonstrate that popular initialization methods often perform poorly and that there are in fact strong alternatives to these methods.

1. Introduction

Clustering, the unsupervised classification of patterns into groups, is one of the most important tasks in exploratory data analysis (Jain, Murty, & Flynn, 1999). Primary goals of clustering include gaining insight into data (detecting anomalies, identifying salient features, etc.), classifying data, and compressing data. Clustering has a long and rich history in a variety of scientific disciplines including anthropology, biology, medicine, psychology, statistics, mathematics, engineering, and computer science. As a result, a plethora of clustering algorithms have been proposed since the early 1950s (Jain, 2010).

Clustering algorithms can be broadly classified into two groups: hierarchical and partitional (Jain, 2010). Hierarchical algorithms recursively find nested clusters either in a top-down (divisive) or bottom-up (agglomerative) fashion. In contrast, partitional algorithms find all the clusters simultaneously as a partition of the data and do not impose a hierarchical structure. Most hierarchical algorithms have quadratic or higher complexity in the number of data points (Jain et al., 1999) and therefore are not suitable for large data sets, whereas partitional algorithms often have lower complexity.

Given a data set $\mathcal{X} = \{\mathbf{x}_1, \mathbf{x}_2, \ldots, \mathbf{x}_N\}$ in $\mathbb{R}^D$, i.e. N points (vectors) each with D attributes (components), hard partitional algorithms divide $\mathcal{X}$ into K exhaustive and mutually exclusive clusters $\mathcal{P} = \{P_1, P_2, \ldots, P_K\}$, $\bigcup_{i=1}^{K} P_i = \mathcal{X}$, $P_i \cap P_j = \emptyset$ for $1 \leq i \neq j \leq K$. These algorithms usually generate clusters by optimizing a criterion function. The most intuitive and frequently used criterion function is the Sum of Squared Error (SSE) given by:

$$\mathrm{SSE} = \sum_{i=1}^{K} \sum_{\mathbf{x}_j \in P_i} \|\mathbf{x}_j - \mathbf{c}_i\|_2^2 \tag{1}$$

where $\|\cdot\|_2$ denotes the Euclidean ($\mathcal{L}_2$) norm and $\mathbf{c}_i = 1/|P_i| \sum_{\mathbf{x}_j \in P_i} \mathbf{x}_j$ is the centroid of cluster P_i whose cardinality is $|P_i|$. The optimization of (1) is often referred to as the minimum SSE clustering (MSSC) problem.

The number of ways in which a set of N objects can be partitioned into K non-empty groups is given by Stirling numbers of the second kind:

$$\mathcal{S}(N,K) = \frac{1}{K!} \sum_{i=0}^{K} (-1)^{K-i} \binom{K}{i} i^N \tag{2}$$

which can be approximated by $K^N/K!$ It can be seen that a complete enumeration of all possible clusterings to determine the global minimum of (1) is clearly computationally prohibitive except for very small data sets (Kaufman & Rousseeuw, 1990). In fact, this non-convex optimization problem is proven to be NP-hard even for $K = 2$ (Aloise, Deshpande, Hansen, & Popat, 2009) or $D = 2$ (Mahajan, Nimbhorkar, & Varadarajan, 2012). Consequently, various heuristics

This Module's Learning Objectives

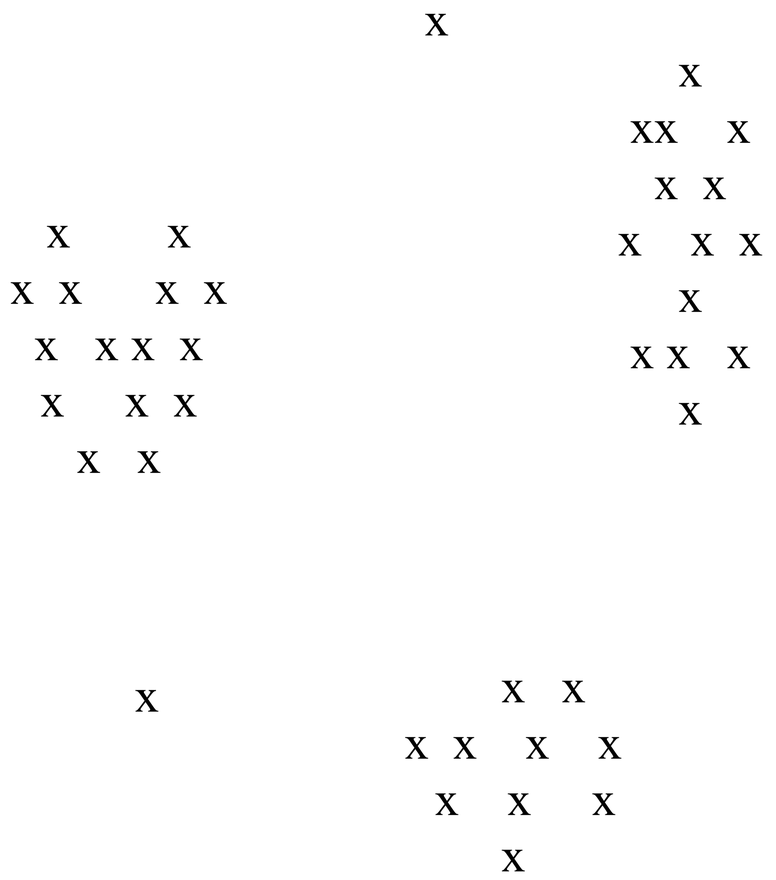
Week 2

Explain how the k-means clustering algorithm identifies clusters

Describe at least two strategies for initializing clusters in k-means

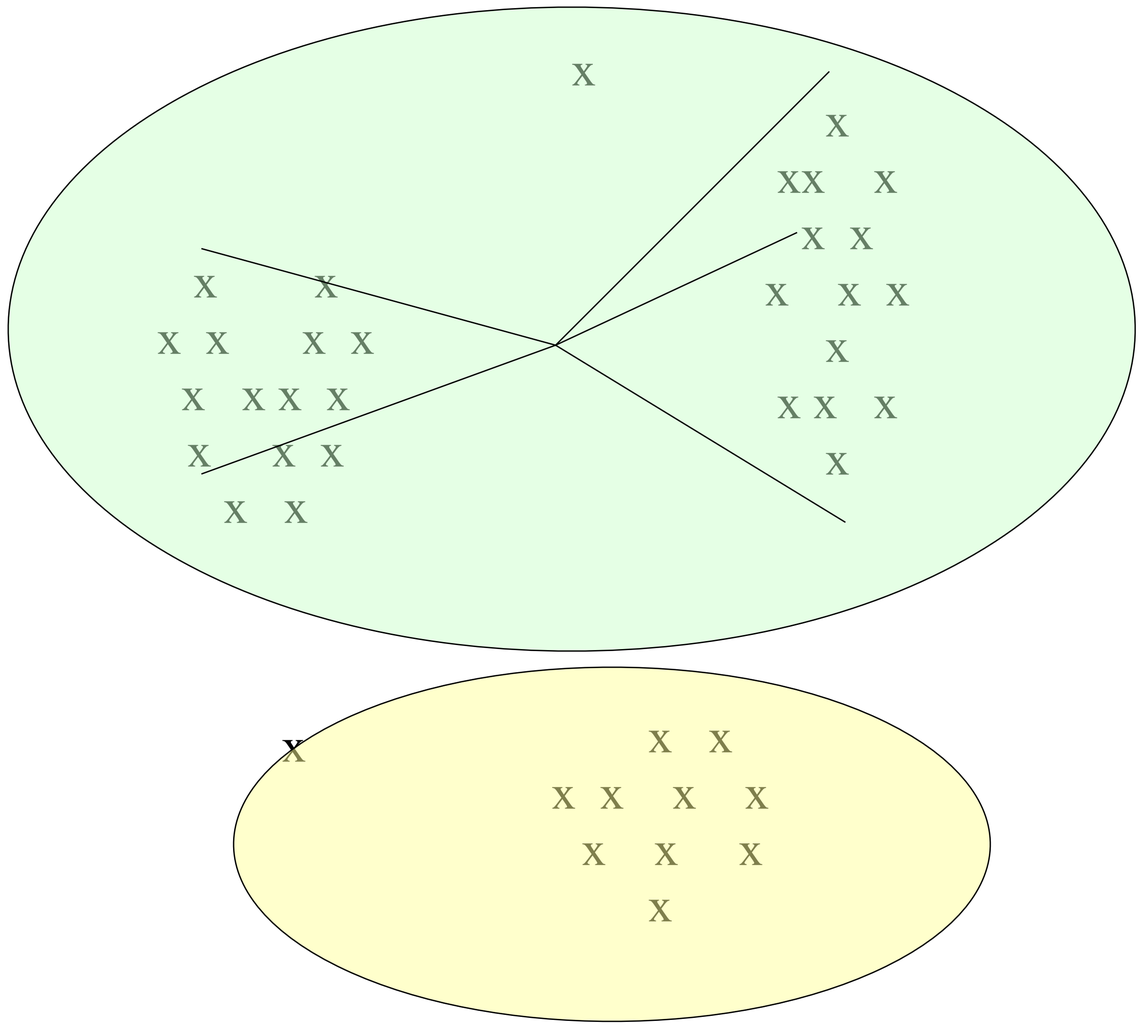
Describe at least two strategies for selecting the number of clusters

Example: Picking *k*



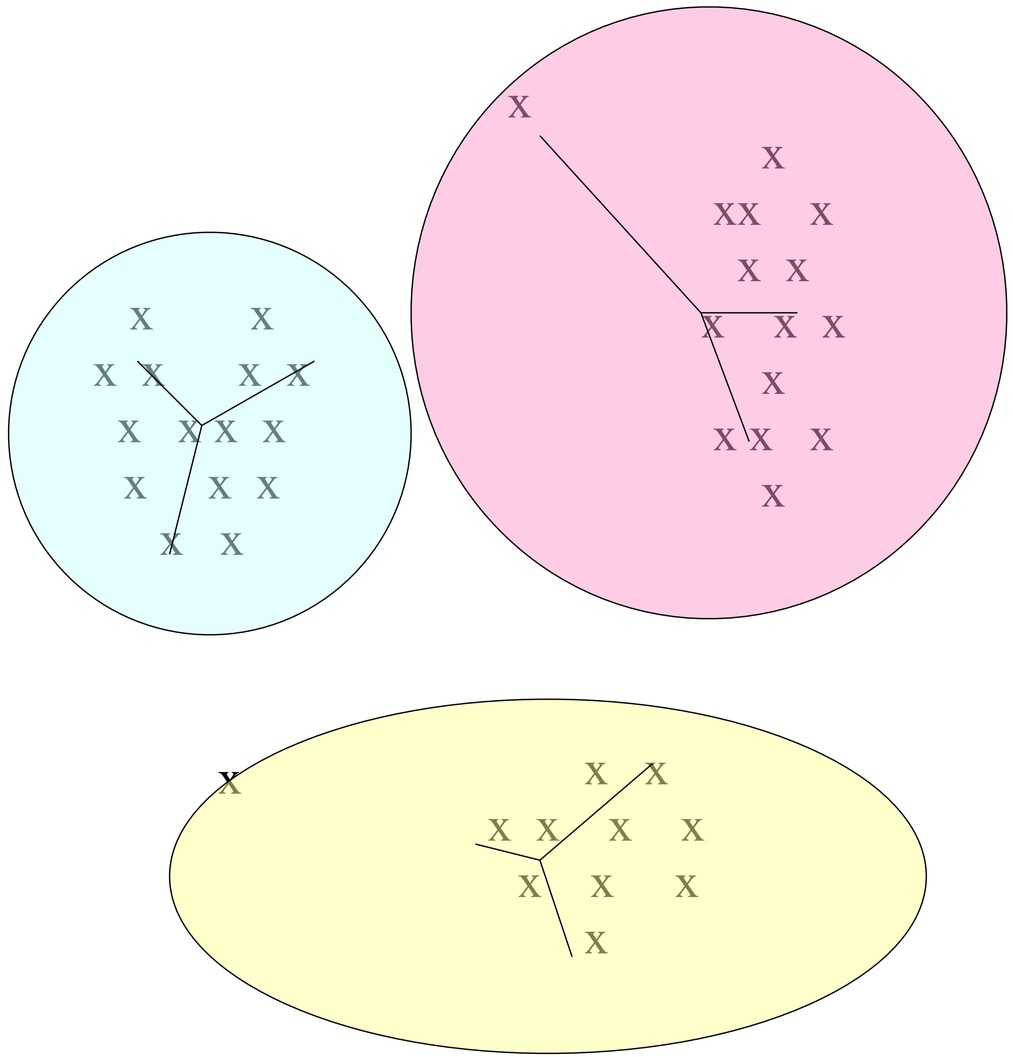
Example: Picking *k*

Too few;
many long
distances
to centroid.



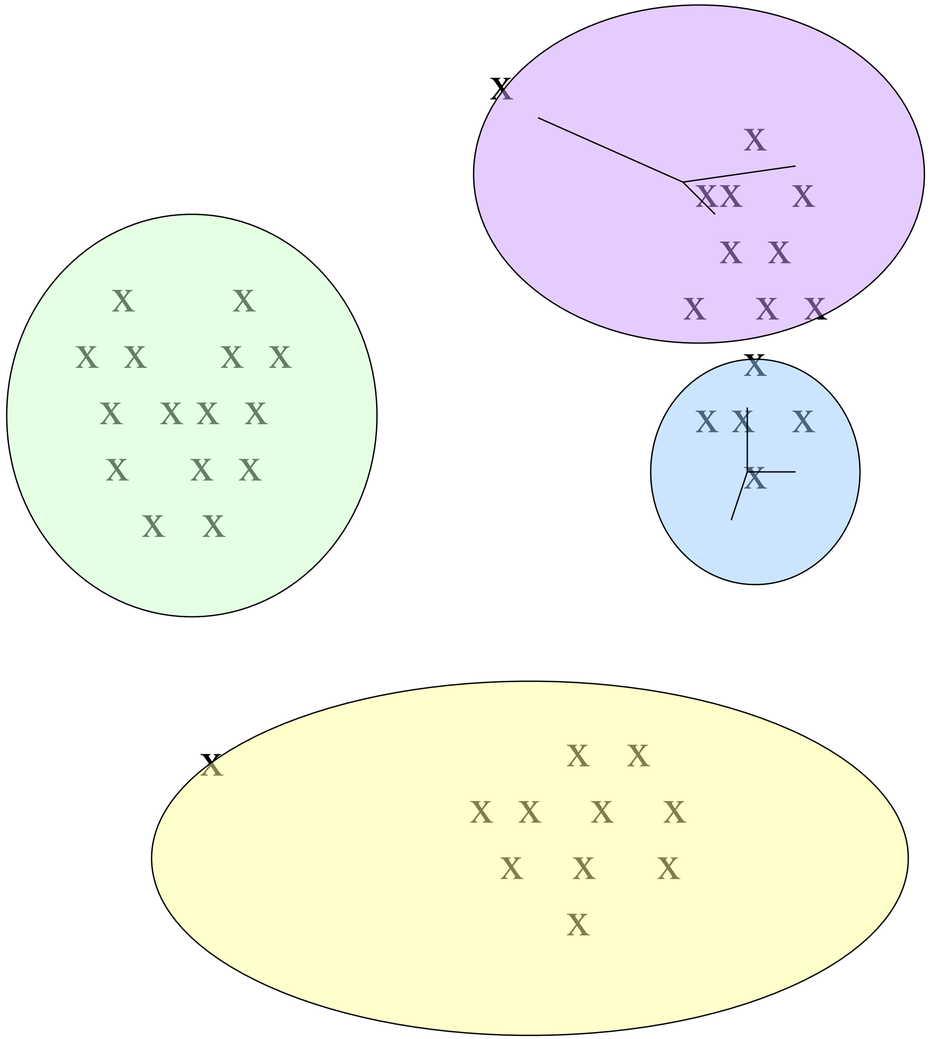
Example: Picking *k*

Just right; distances rather short.

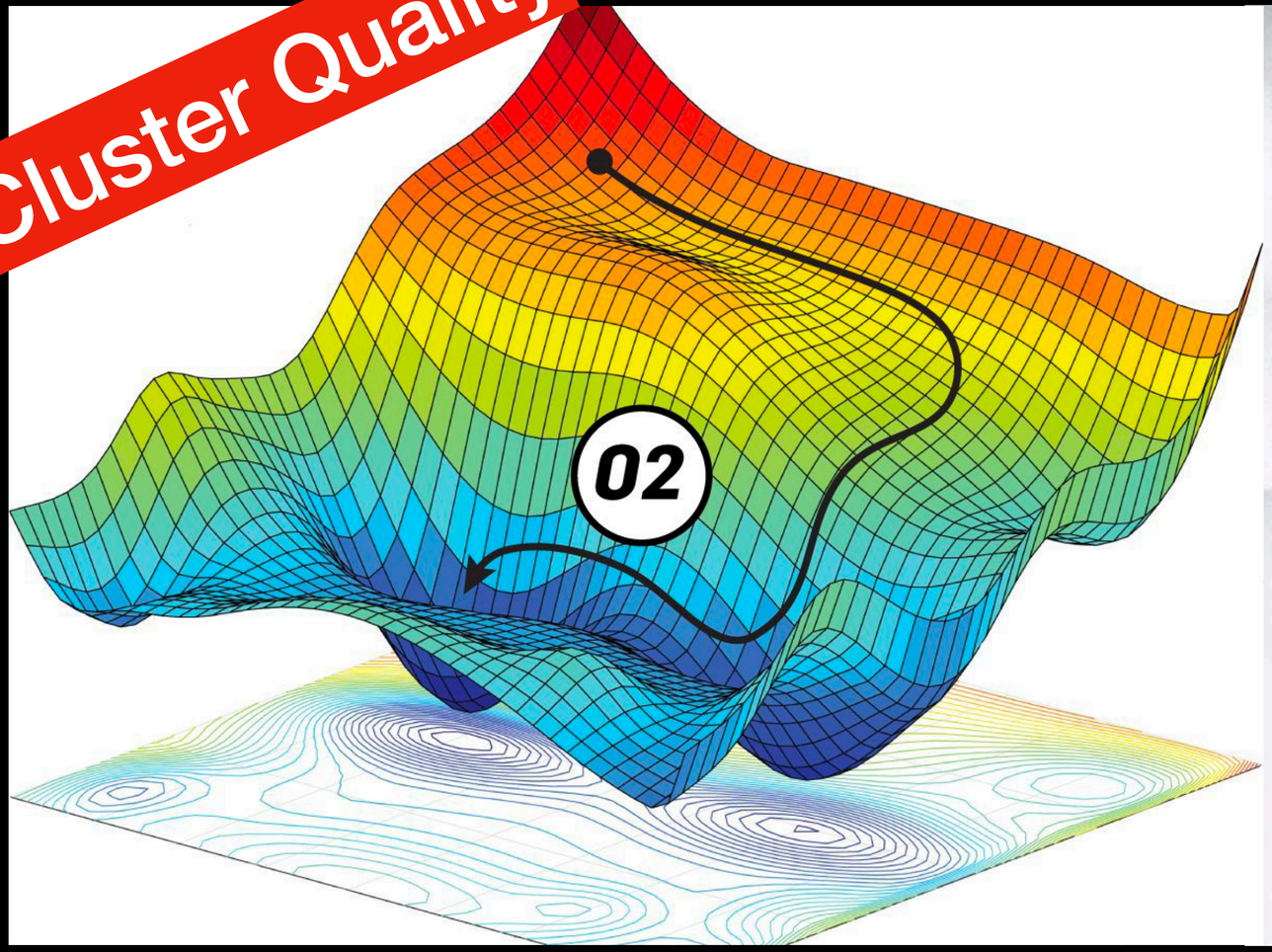


Example: Picking *k*

Too many;
little improvement
in average
distance.



Cluster Quality

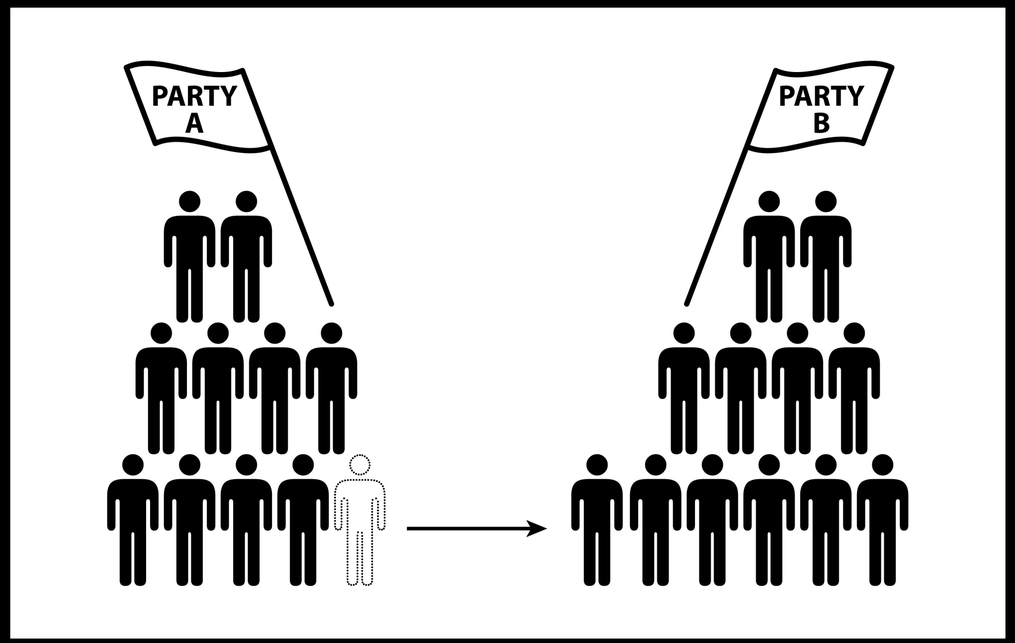


By fiat, specify *k* value

Or you specify a **termination condition**

Given a value for ***k***, split/merge until you have ***k*** clusters

Where does ***k*** come from?



Maybe you know from domain knowledge

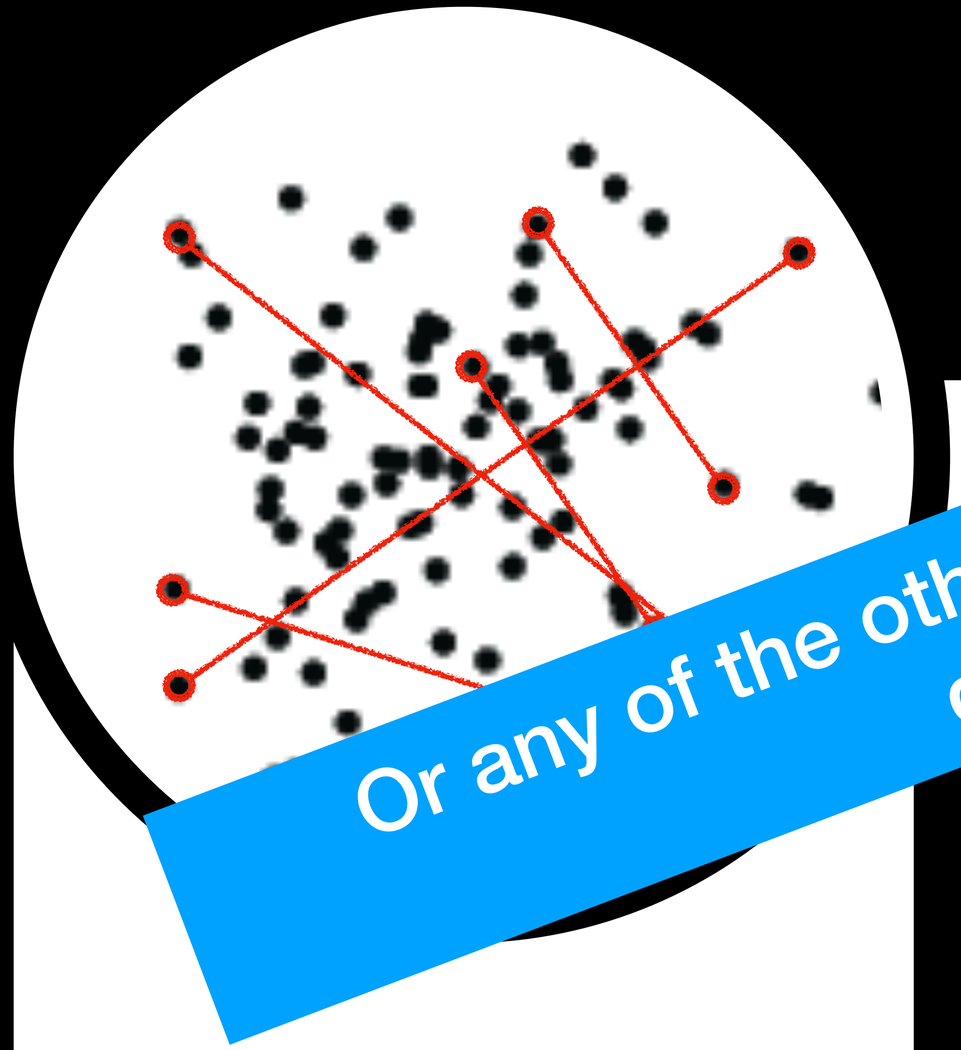
Or some other knowledge or restriction

How do you choose a ***k*** value?

Great question

TBD

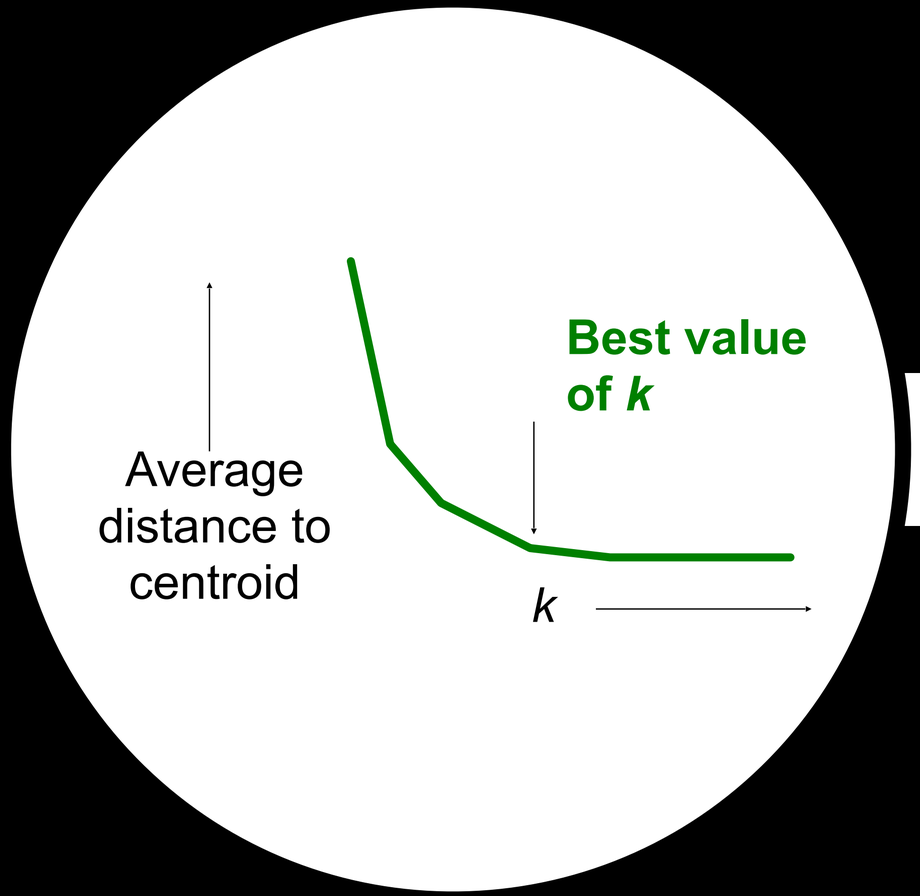
Common metric for clustering is cohesion



Or any of the other cohesion metrics we discussed

...hesion via Avg. Distance

Minimize the average distance between any two points in the cluster



Select multiple values for ***k***…

For each one, run clustering until termination

Measure cohesion at termination

Choose ***k*** where metric levels off

Determining the number of clu

en.wikipedia.org/wiki/Determining_the_number_of_clusters_in_a_data_set

Apps | Jordan's Style | https://inf.ethz.ch/... | tome - Synology... | Other Bookmarks

Contribute
Help
Learn to edit
Community portal
Recent changes
Upload file

Tools
What links here
Related changes
Special pages
Permanent link
Page information
Cite this page
Wikidata item

Print/export
Download as PDF
Printable version

Languages
فارسی
Edit links

The correct choice of *k* is often ambiguous, with interpretations depending on the shape and scale of the distribution of points in a data set and the desired clustering resolution of the user. In addition, increasing *k* without penalty will always reduce the amount of error in the resulting clustering, to the extreme case of zero error if each data point is considered its own cluster (i.e., when *k* equals the number of data points, *n*). Intuitively then, *the optimal choice of* k *will strike a balance between maximum compression of the data using a single cluster, and maximum accuracy by assigning each data point to its own cluster.* If an appropriate value of *k* is not apparent from prior knowledge of the properties of the data set, it must be chosen somehow. There are several categories of methods for making this decision.

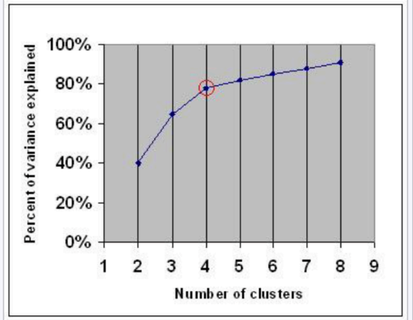
Contents [hide]

The elbow method [edit]

Main article: Elbow method (clustering)

The elbow method looks at the percentage of variance explained as a function of the number of clusters: One should choose a number of clusters so that adding another cluster doesn't give much better modeling of the data. More precisely, if one plots the percentage of variance explained by the clusters against the number of clusters, the first clusters will add much information (explain a lot of variance), but at some point the marginal gain will drop, giving an angle in the graph. The number of clusters is chosen at this point, hence the "elbow criterion". This "elbow" cannot always be unambiguously identified,[1] making this method very subjective and unreliable. Percentage of variance explained is the ratio of the between-group variance to the total variance, also known as an F-test. A slight variation of this method plots the curvature of the within group variance.[2]

The method can be traced to speculation by Robert L. Thorndike in 1953.[3]



Explained Variance. The "elbow" is indicated by the red circle. The number of clusters chosen should therefore be 4.

X-means clustering [edit]

In statistics and data mining, **X-means clustering** is a variation of k-means clustering that refines cluster assignments by repeatedly attempting subdivision, and keeping the best resulting splits, until a criterion

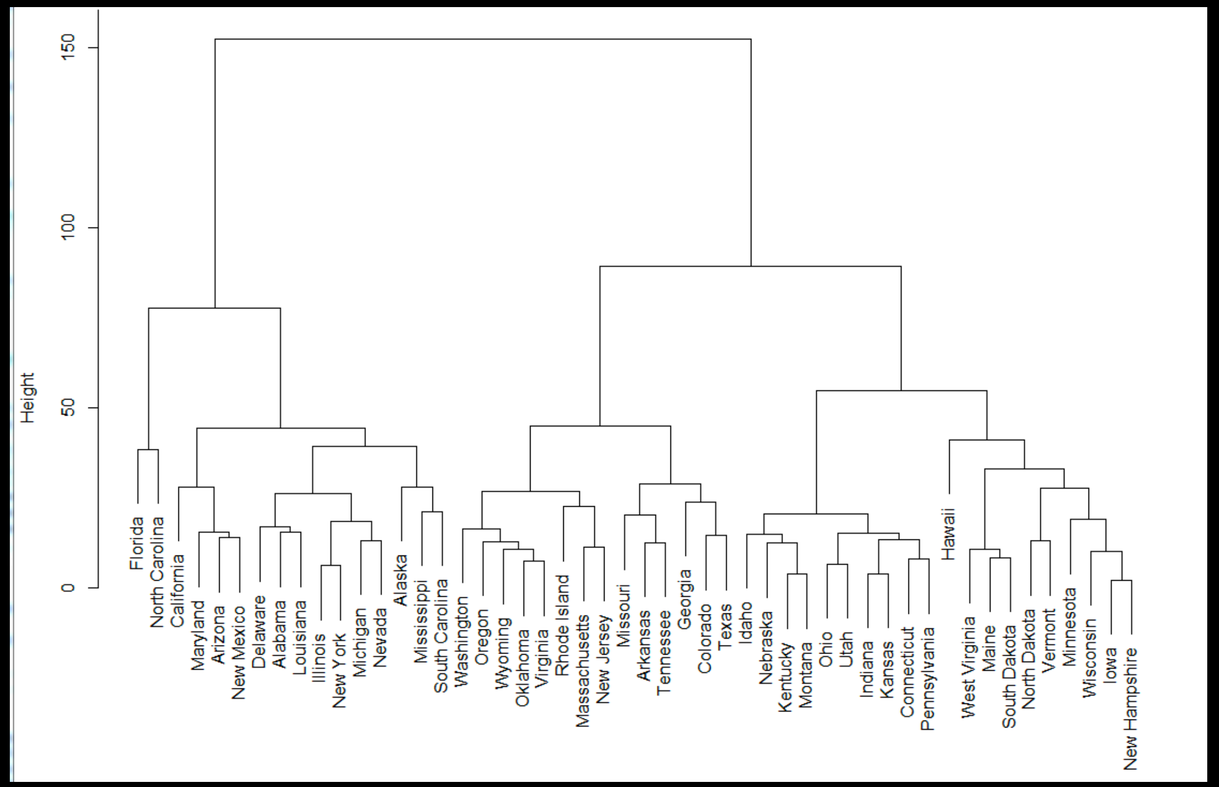
Often referred to as the "Elbow Method"

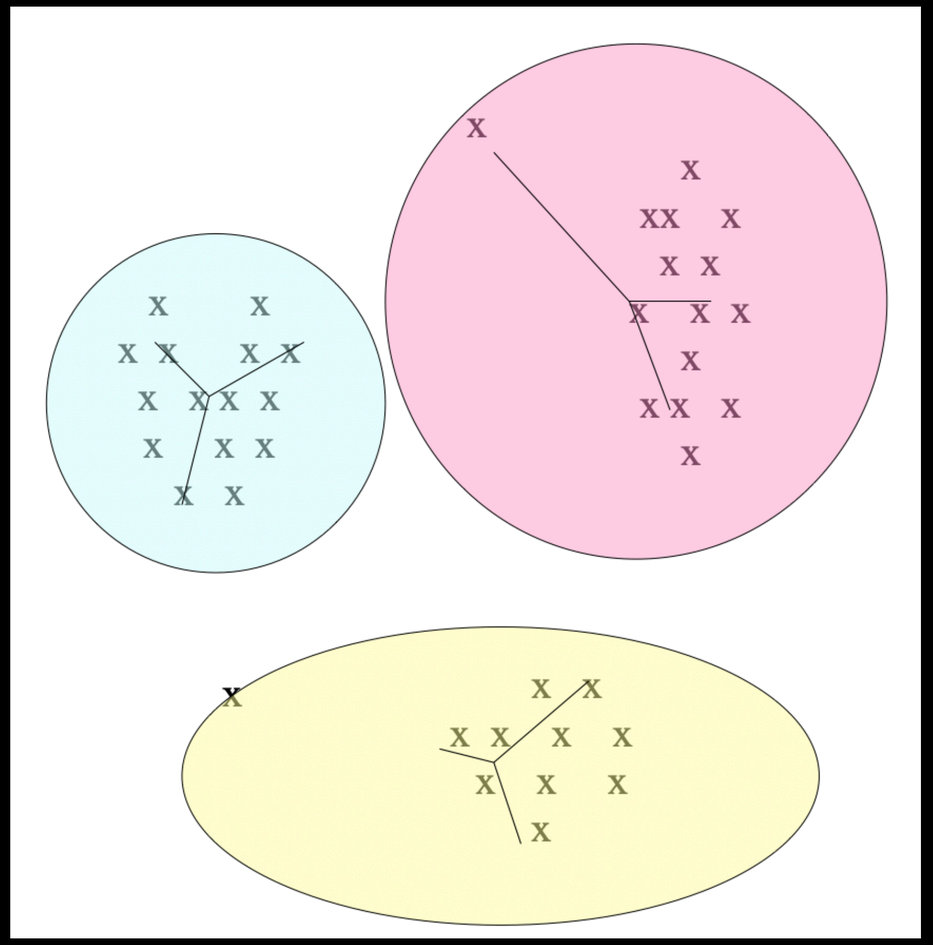
A third option…

Performance in a downstream task

How does stopping differ in k-means and in hierarchical cluster?

k tells when to stop merging





We <u>always</u> have *k* clusters, and stop after convergence

This Module's Learning Objectives

Part 2

- Explain how the k-means clustering algorithm identifies clusters
- Describe at least two strategies for initializing clusters in k-means
- Describe at least two strategies for selecting the number of clusters

Questions?

Prof. Cody Buntain | @codybuntain | cbuntain@umd.edu
Director, Information Ecosystems Lab