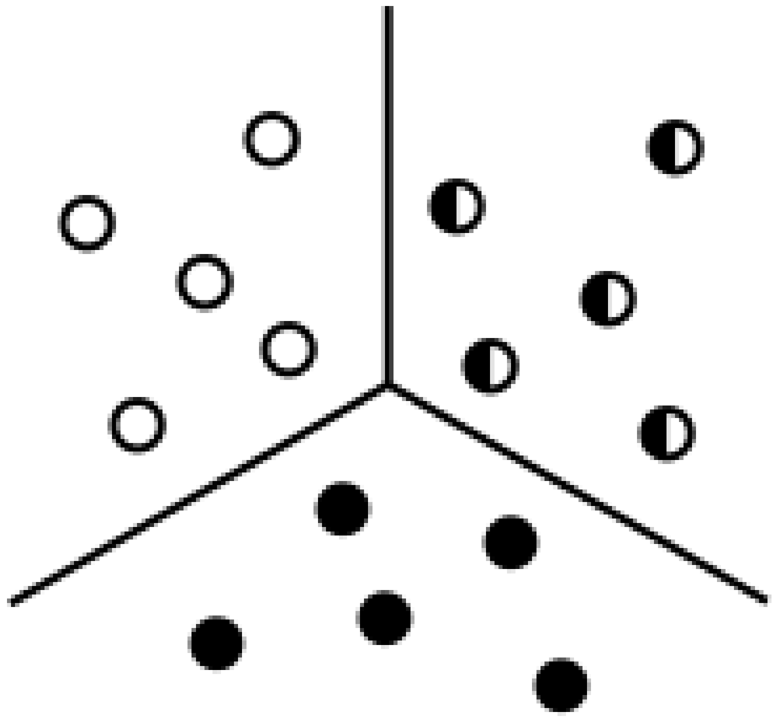
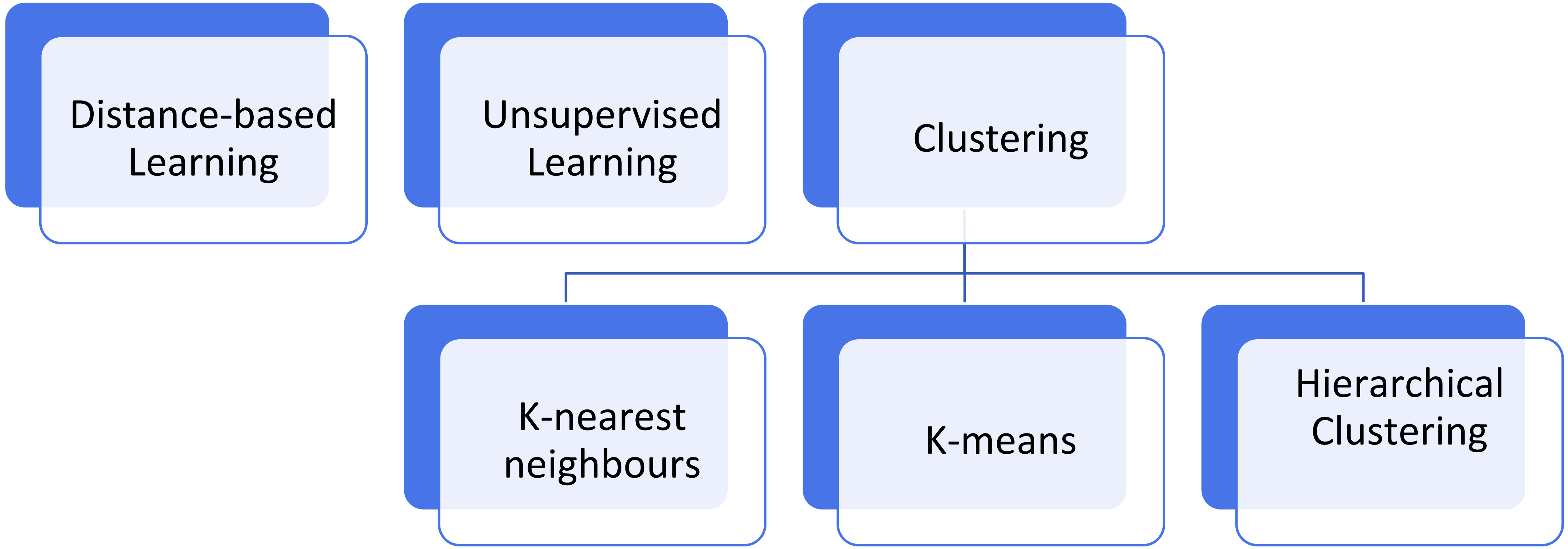


Clustering


Distance-based Learning
Unsupervised Learning
Clustering
K-nearest neighbours
K-means
Hierarchical Clustering

Distance-based Learning

Inputs:

- Data set
- Notion of distance/closeness between instances

Outputs:

- Classifier, regressor
- “Structure” in the data, e.g. groups of similar points
- Result is based on notion of “closeness” or “distance” provided

Unsupervised Learning

Only features $x_1, \ldots, x_p$

None is more important than the others

Discover relationships among the instances and features

Difficult to evaluate “performance” because task is ill-defined

You might use the output of unsupervised learning for supervised learning

Often, but not always, “distance based”

Examples: clustering (k-means, DBSCAN, …), dimensionality reduction (PCA, isomap, …)

K-nearest neighbours in 1 slide

Supervised learning

Take your training data $(\mathbf{x}, y)$ pairs…

- … and do nothing [sometimes called a *lazy learner*]

To predict $\hat{y}$ for new $\mathbf{x}$, find the K "closest" $\mathbf{x}$s in the training data, and summarize their ys

- Average
- Majority class
- Proportion
- Whatever

How to choose K? Choose definition of "closest"?

K-nn Decision Boundary

K-Means Clustering

What is clustering?

Clustering is grouping “similar” objects together.

- To establish prototypes, or detect outliers.
- To simplify data for further analysis/learning.
- To visualize data (in conjunction with dimensionality reduction)

The notion of similarity must be given to the clustering algorithm.

Clusterings are not “right” or “wrong” – different clusterings/clustering criteria can reveal different things about the data.

Clustering Algorithms

Clustering algorithms:

- Employ some notion of similarity between objects
- Have an explicit or implicit criterion defining what a good cluster is
- Heuristically optimize that criterion to determine the clustering

Some clustering criteria/algorithms have natural probabilistic interpretations

K-means clustering

One of the most commonly-used clustering algorithms, because it is easy to implement and quick to run.

Assumes the objects (instances) to be clustered are p-dimensional vectors, $\mathbf{x}_i$.

Uses a distance measure between the instances (typically Euclidean distance)

The goal is to *partition* the data into K disjoint subsets

Loss Function

Given:

- A set of feature vectors $\{\mathbf{x}_1, \mathbf{x}_2, \dots, \mathbf{x}_n\}$
- K disjoint sets $C_1, \dots, C_K$ such that C_i contains the indices of all the features belonging to the ith cluster and such that $\cup_i C_i = \{1, \dots, n\}$
- K *centroids* $\mathbf{c}_1, \dots, \mathbf{c}_K$

Loss function is

$$\sum_{i=1}^{K} \sum_{j \in C_i} \| \mathbf{x}_j - \mathbf{c}_i \|^2$$

Minimizing the Loss

Loss function is

$$\sum_{i=1}^{K} \sum_{j \in C_i} \| \mathbf{x}_j - \mathbf{c}_i \|^2$$

Need to find both the locations of the centroids $\mathbf{c}_i$, and a “good” assignment of each feature vector to a centroid defined by the C_i.

This is an NP-hard problem

Solution: **Expectation – Maximization.**

Minimizing the Loss

Loss function is

$$\sum_{i=1}^{K} \sum_{j \in C_i} \| \mathbf{x}_j - \mathbf{c}_i \|^2$$

If we hold the C_i fixed, what are the best $\mathbf{c}_i$?

Minimizing the Loss

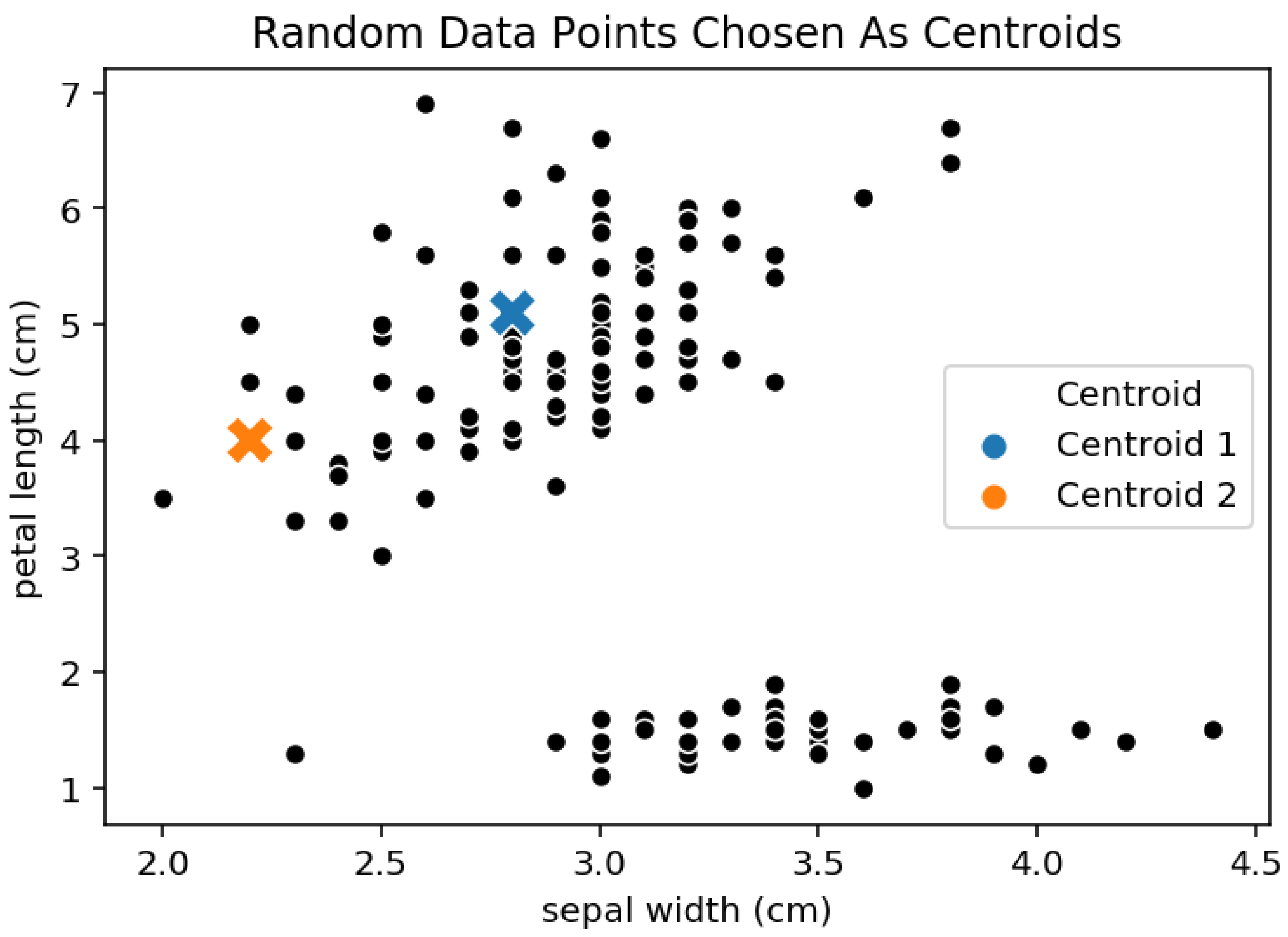
Loss function is

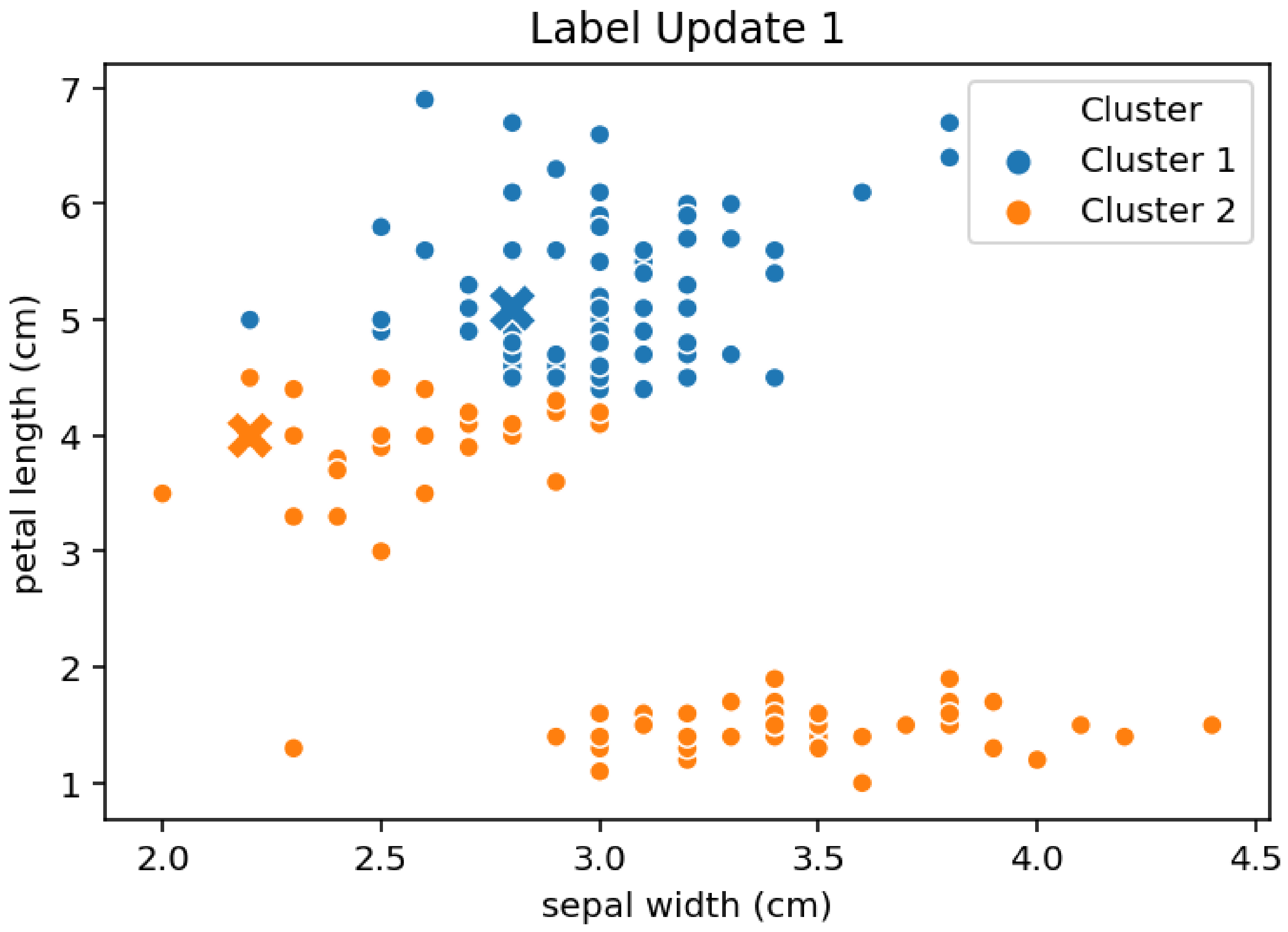
$$\sum_{i=1}^{K} \sum_{j \in C_i} \| \mathbf{x}_j - \mathbf{c}_i \|^2$$

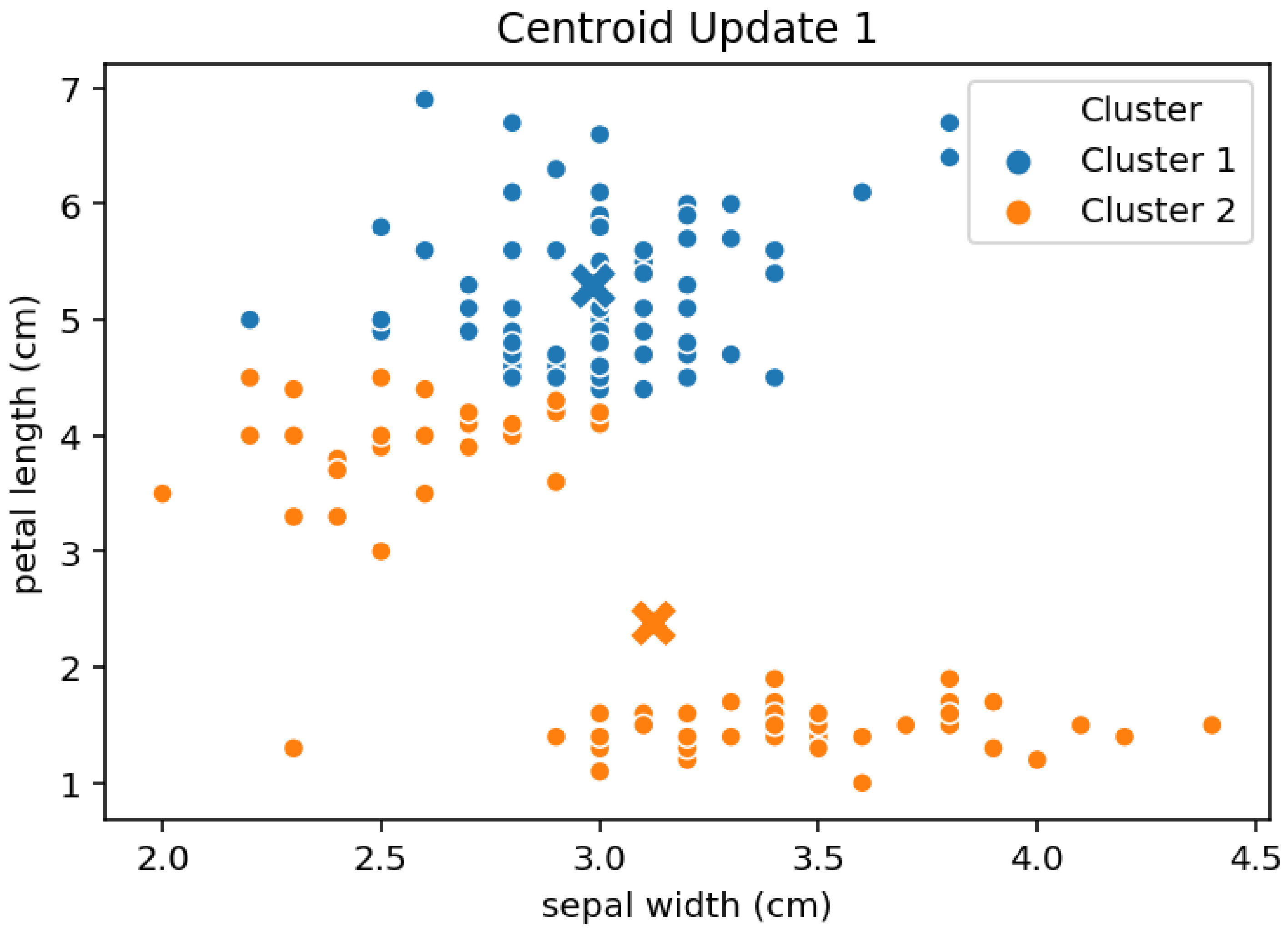
If we hold the $\mathbf{c}_i$ fixed, what are the best C_i?

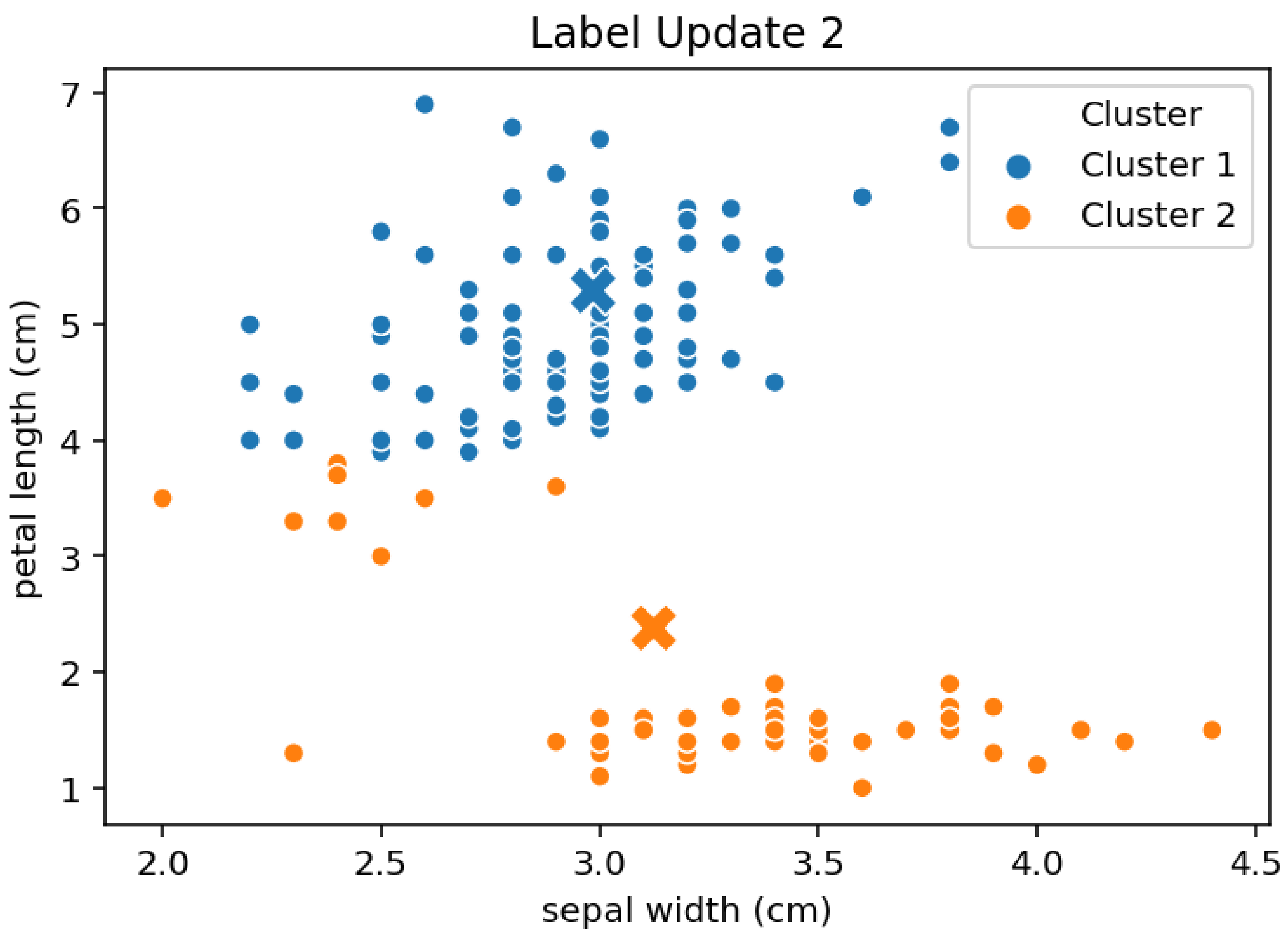
K-means clustering

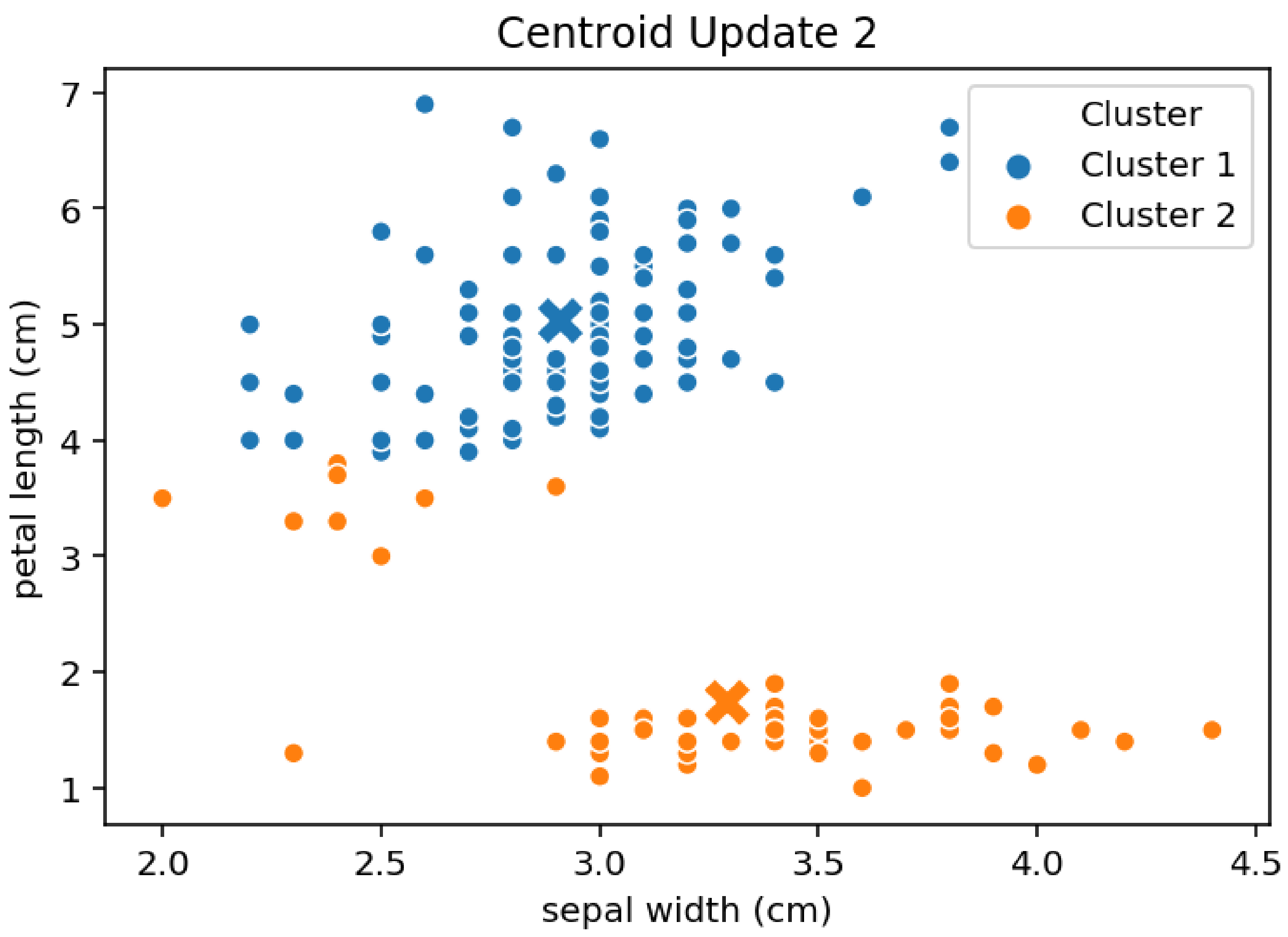
- Inputs:
  - A set of p-dimensional real vectors $\{\mathbf{x}_1, \mathbf{x}_2, \ldots, \mathbf{x}_n\}$.
  - K, the desired number of clusters.
- Output: A mapping of the vectors into K clusters (disjoint subsets), $C: \{1, \ldots, n\} \mapsto \{1, \ldots, K\}$.
- Initialize C randomly.
- Repeat:
  - Compute the *centroid* of each cluster (the mean of all the instances in the cluster)
  - Reassign each instance to the cluster with closest centroid
- until C stops changing.

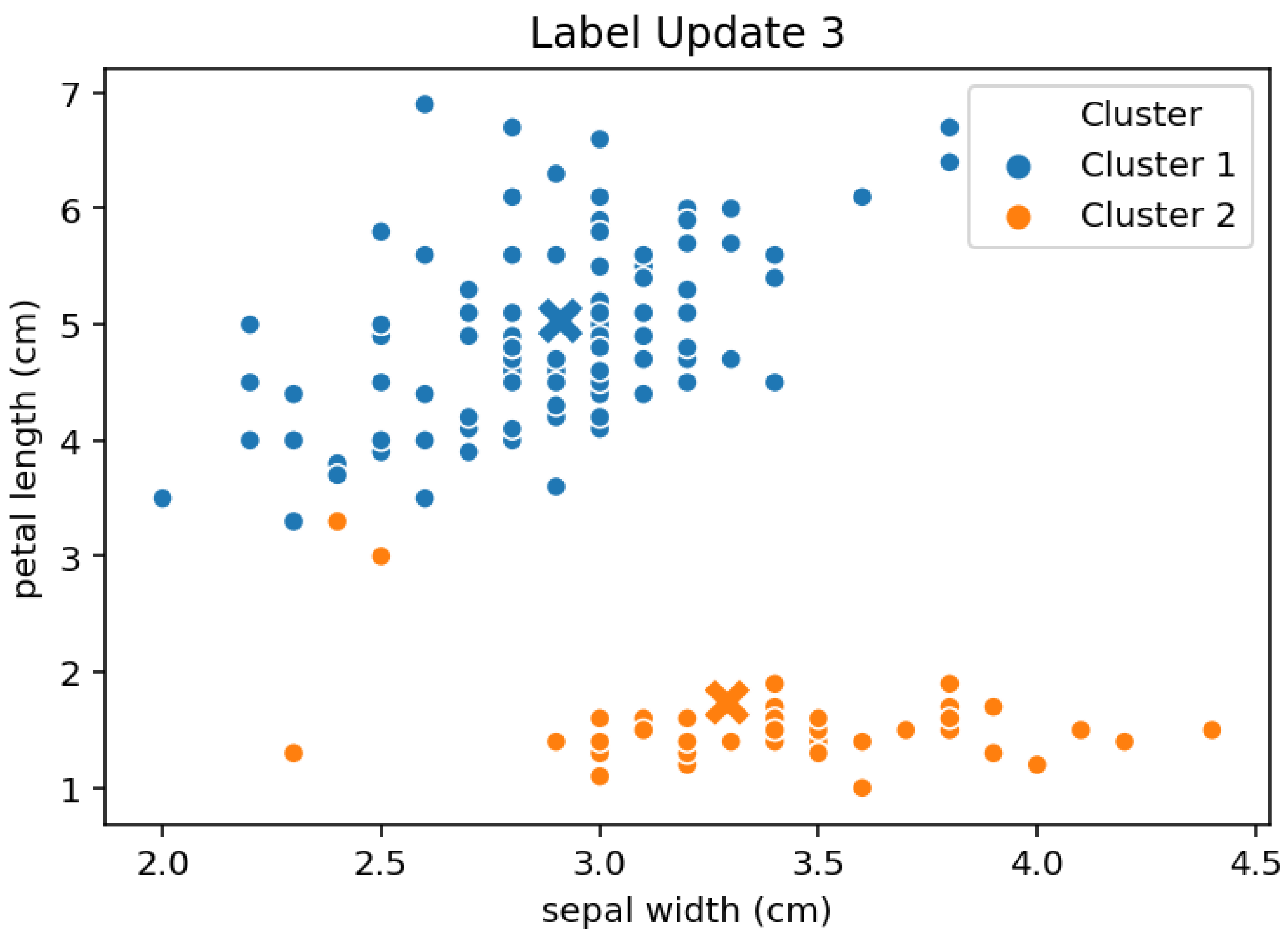

Random Data Points Chosen As Centroids
petal length (cm)
sepal width (cm)
Centroid
Centroid 1
Centroid 2
1
2
3
4
5
6
7
2.0
2.5
3.0
3.5
4.0
4.5

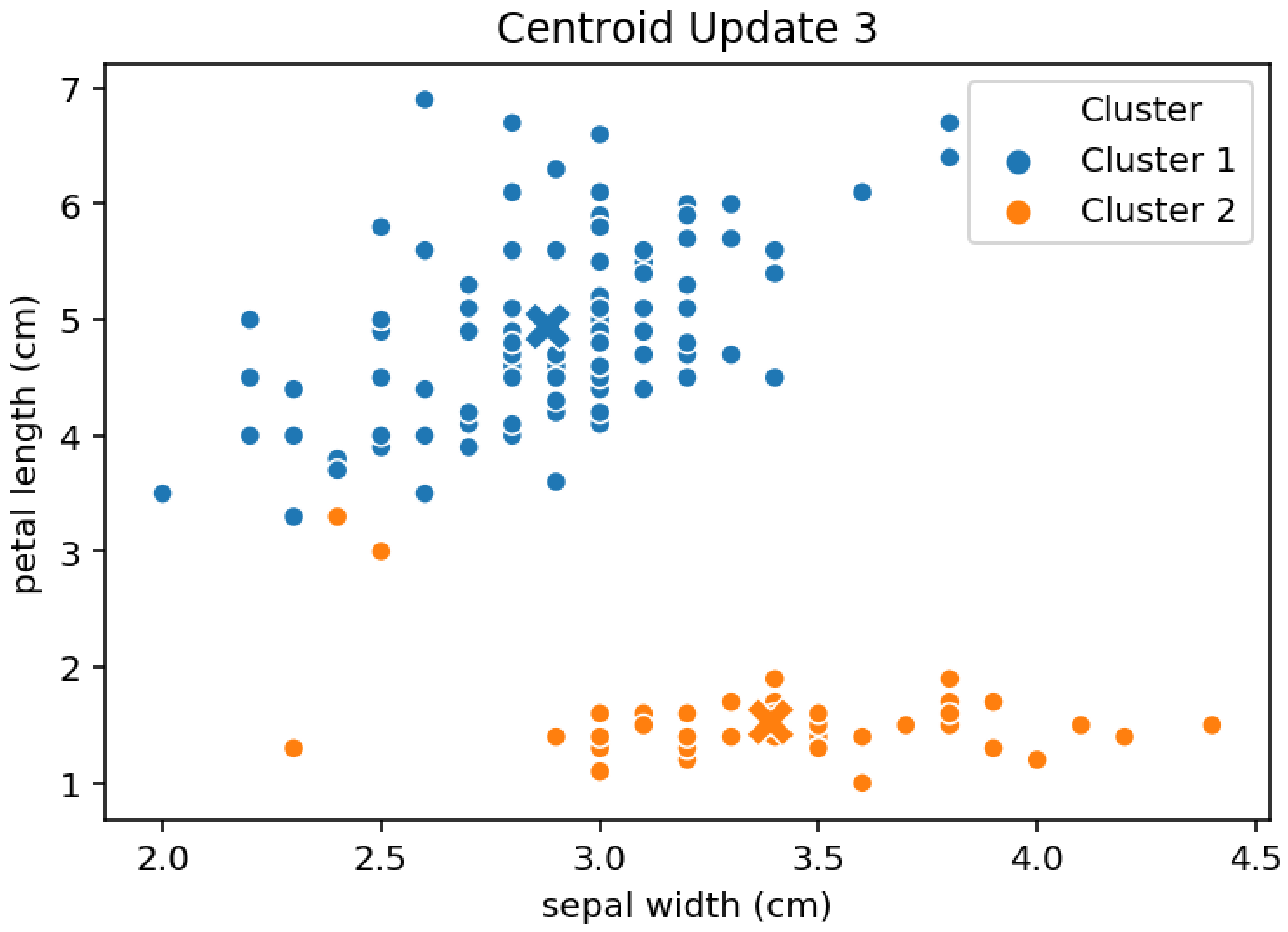

Label Update 1
Cluster
Cluster 1
Cluster 2
petal length (cm)
sepal width (cm)
1
2
3
4
5
6
7
2.0
2.5
3.0
3.5
4.0
4.5

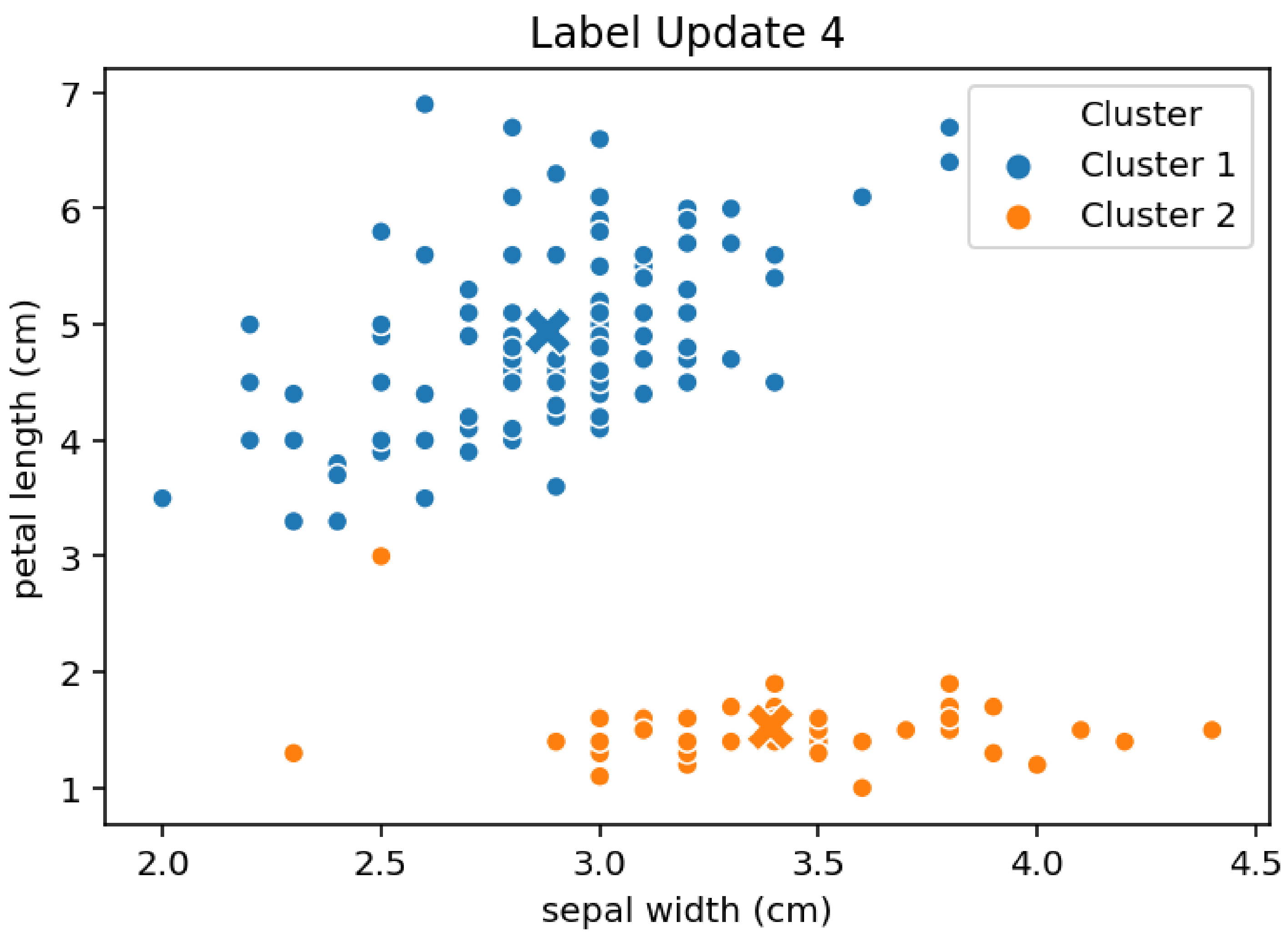

Centroid Update 1
Cluster
Cluster 1
Cluster 2
petal length (cm)
sepal width (cm)
1
2
3
4
5
6
7
2.0
2.5
3.0
3.5
4.0
4.5

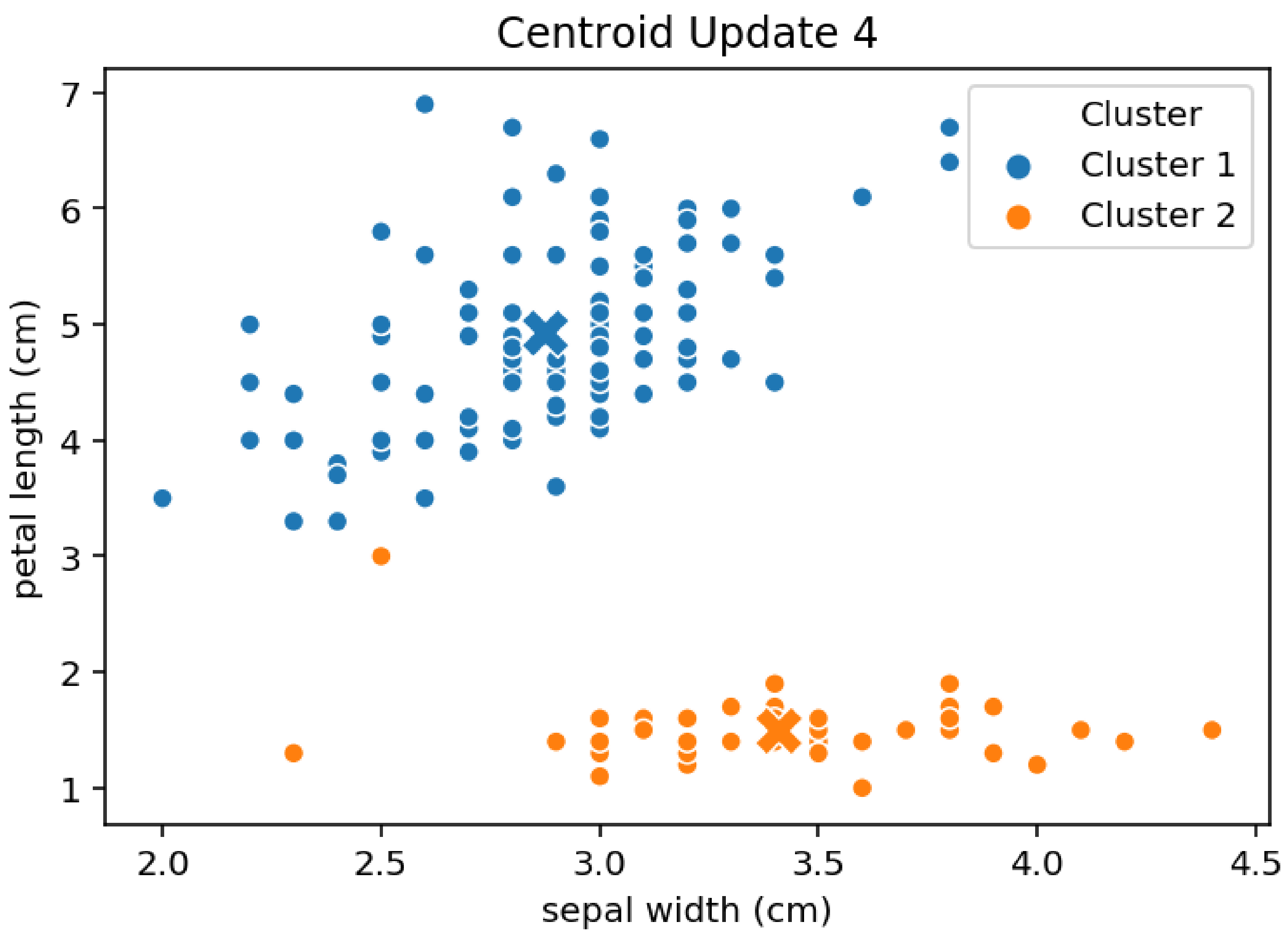

Label Update 2
Cluster
Cluster 1
Cluster 2
petal length (cm)
sepal width (cm)
1
2
3
4
5
6
7
2.0
2.5
3.0
3.5
4.0
4.5

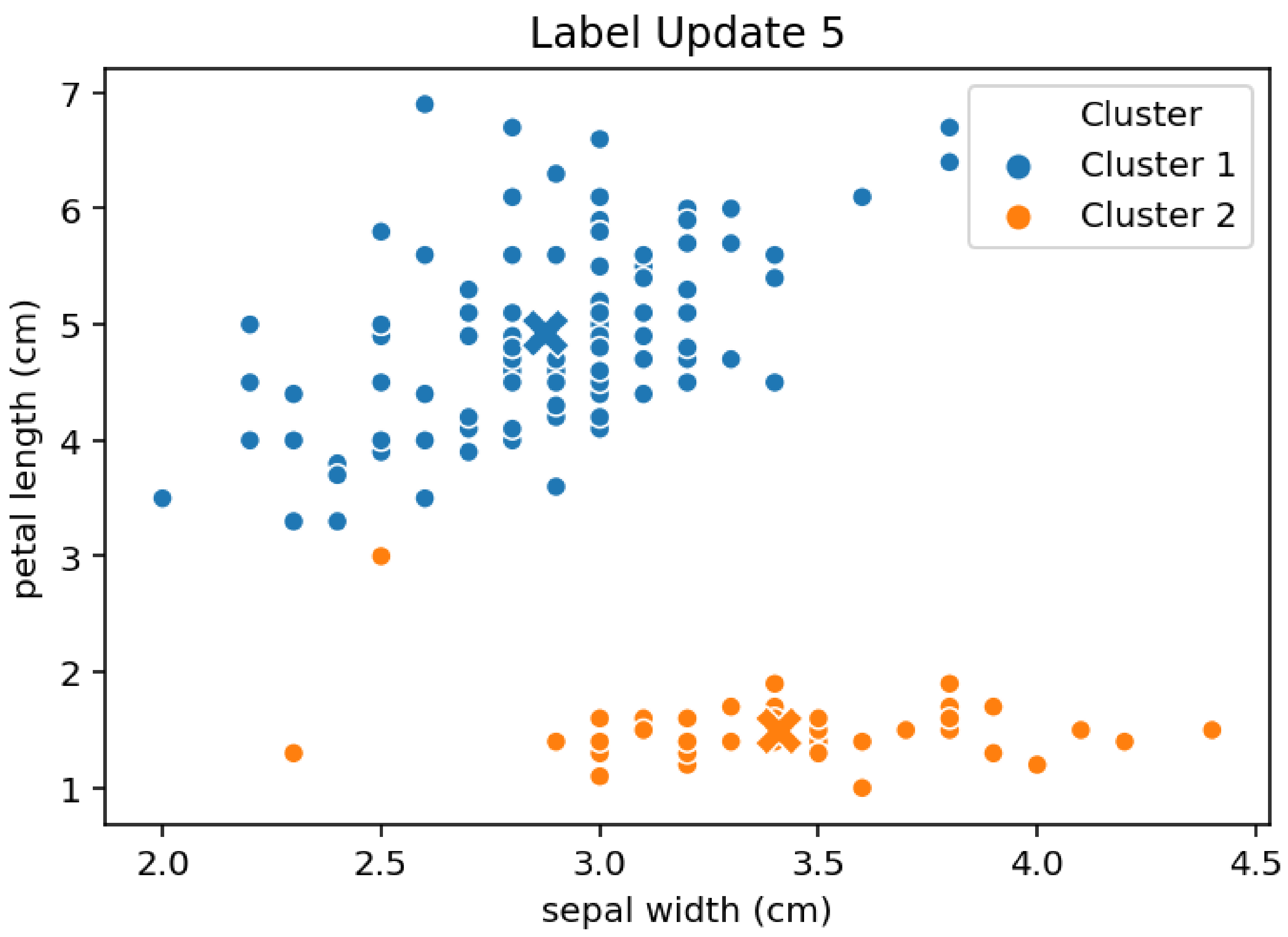

Centroid Update 2
Cluster
Cluster 1
Cluster 2
petal length (cm)
sepal width (cm)
1
2
3
4
5
6
7
2.0
2.5
3.0
3.5
4.0
4.5

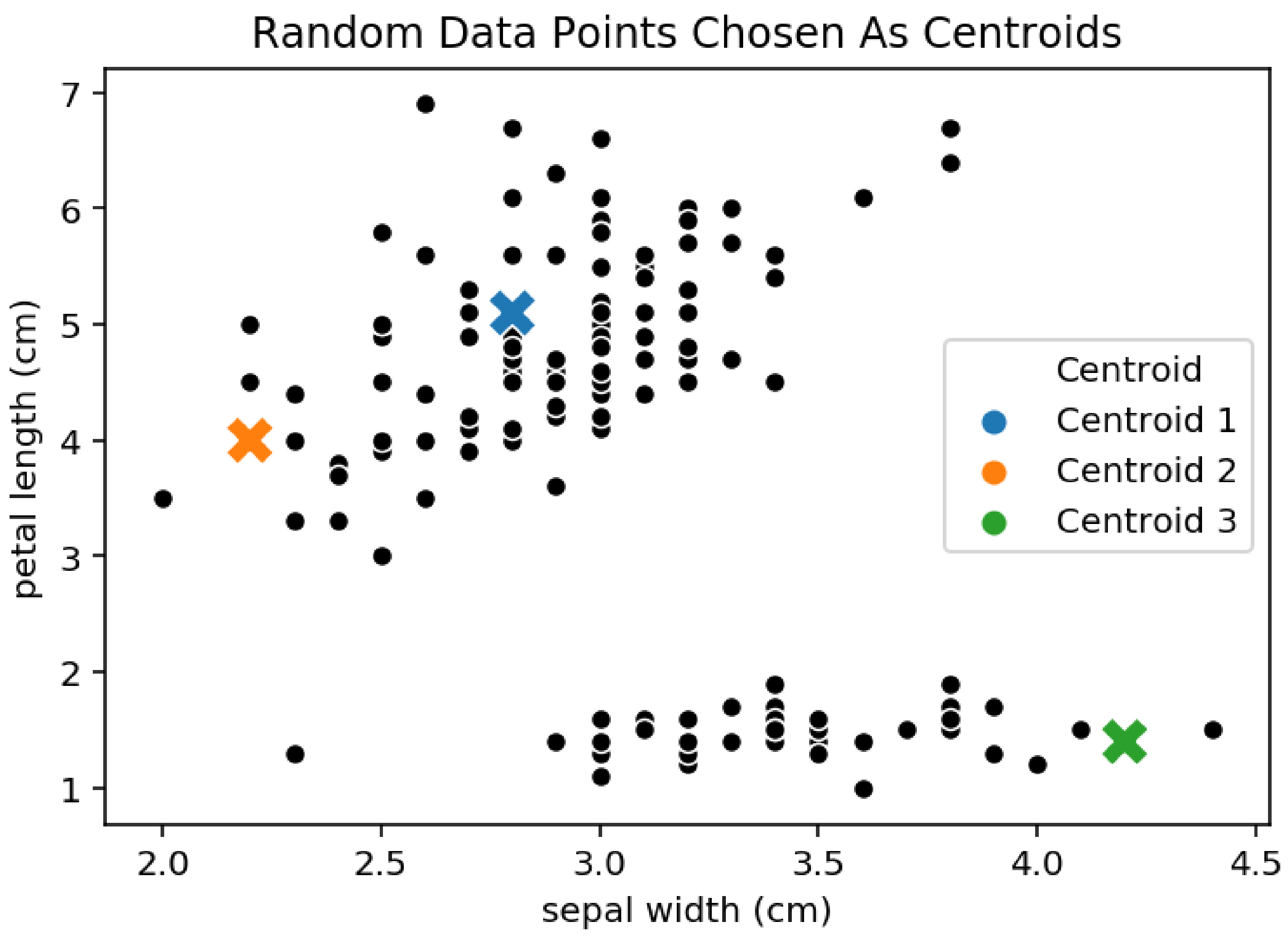

Label Update 3
petal length (cm)
sepal width (cm)
Cluster
Cluster 1
Cluster 2
7
6
5
4
3
2
1
2.0
2.5
3.0
3.5
4.0
4.5

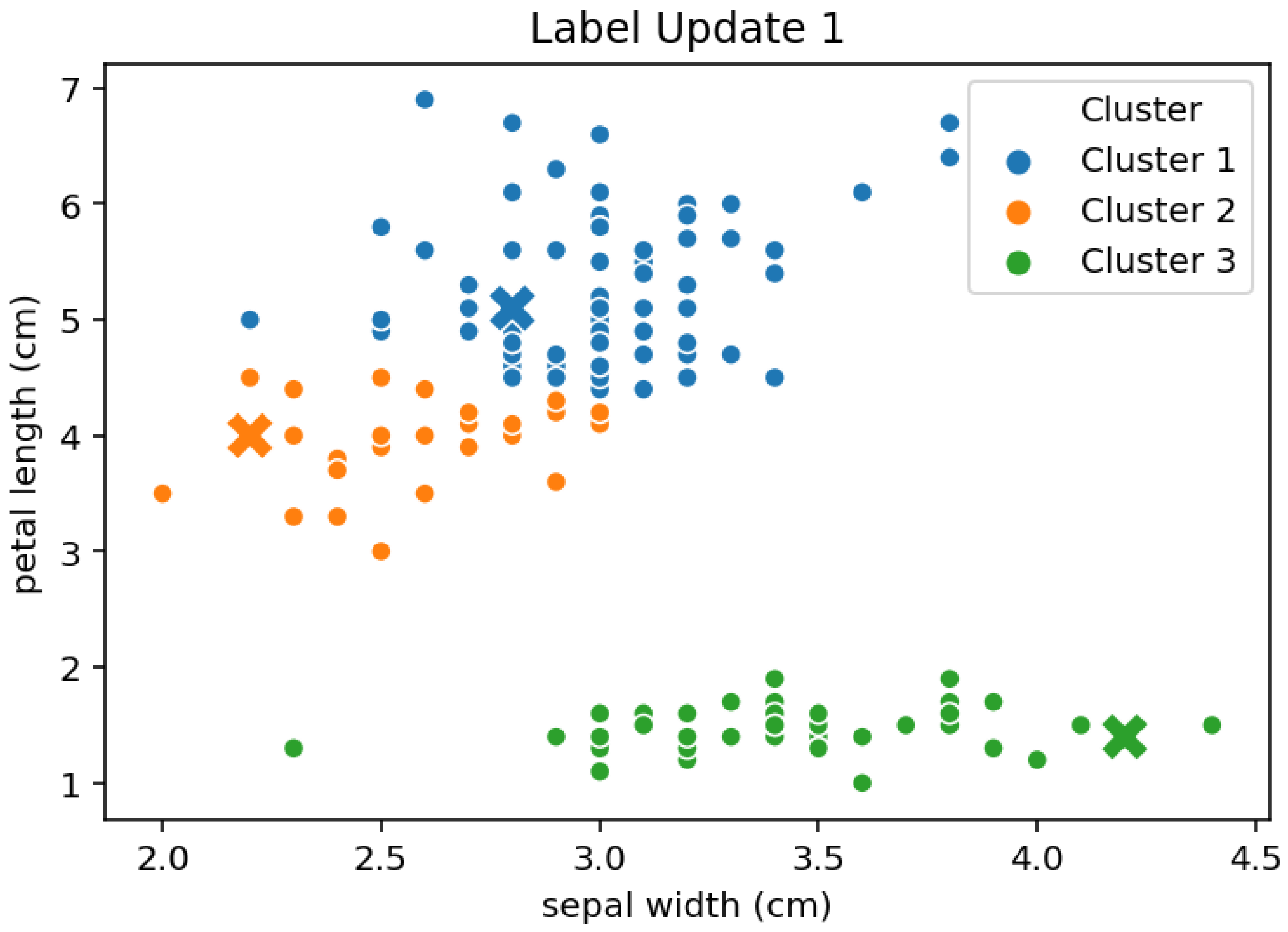

Centroid Update 3
petal length (cm)
sepal width (cm)
Cluster
Cluster 1
Cluster 2
1
2
3
4
5
6
7
2.0
2.5
3.0
3.5
4.0
4.5

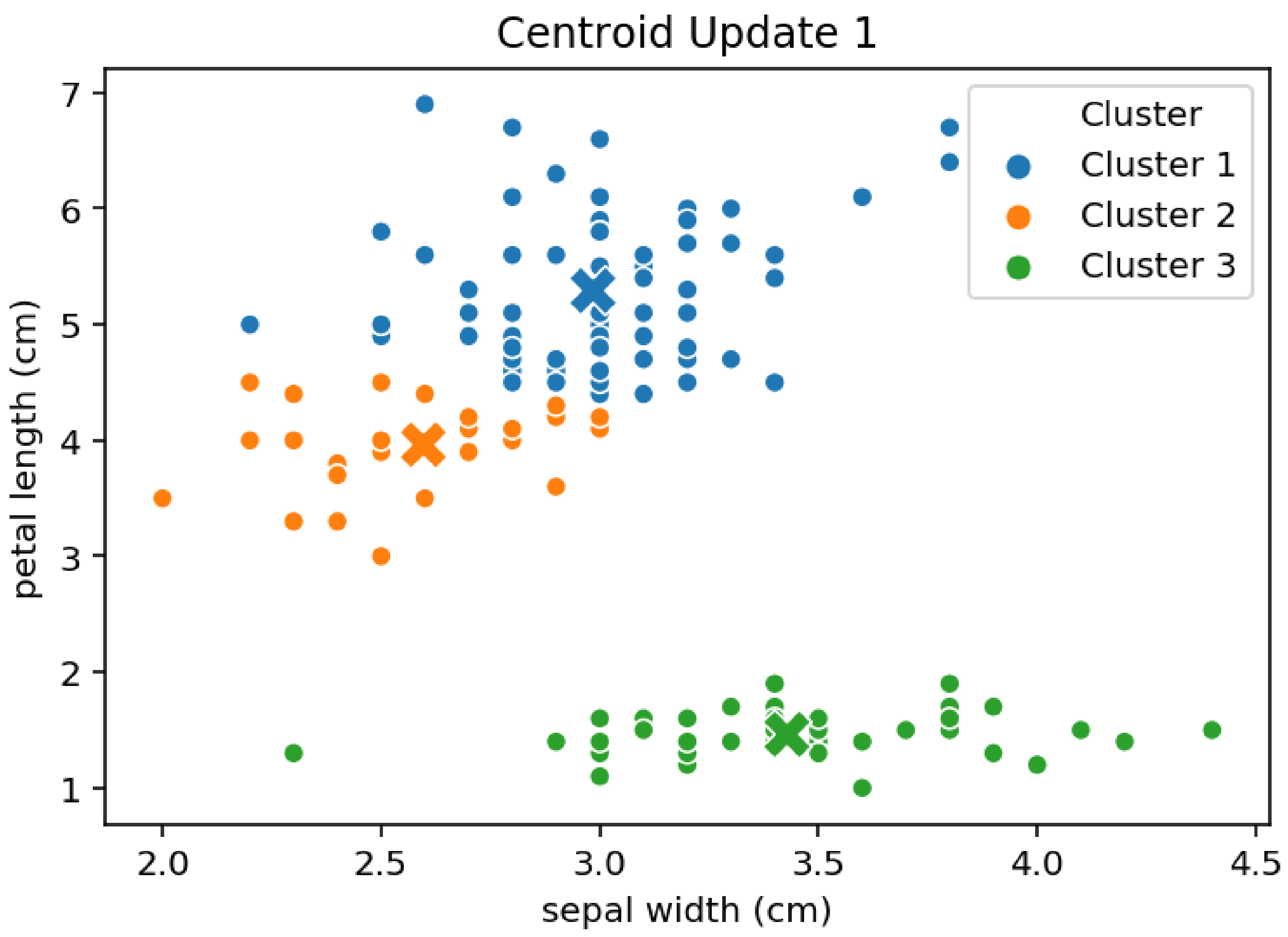

Label Update 4
petal length (cm)
sepal width (cm)
Cluster
Cluster 1
Cluster 2
7
6
5
4
3
2
1
2.0
2.5
3.0
3.5
4.0
4.5

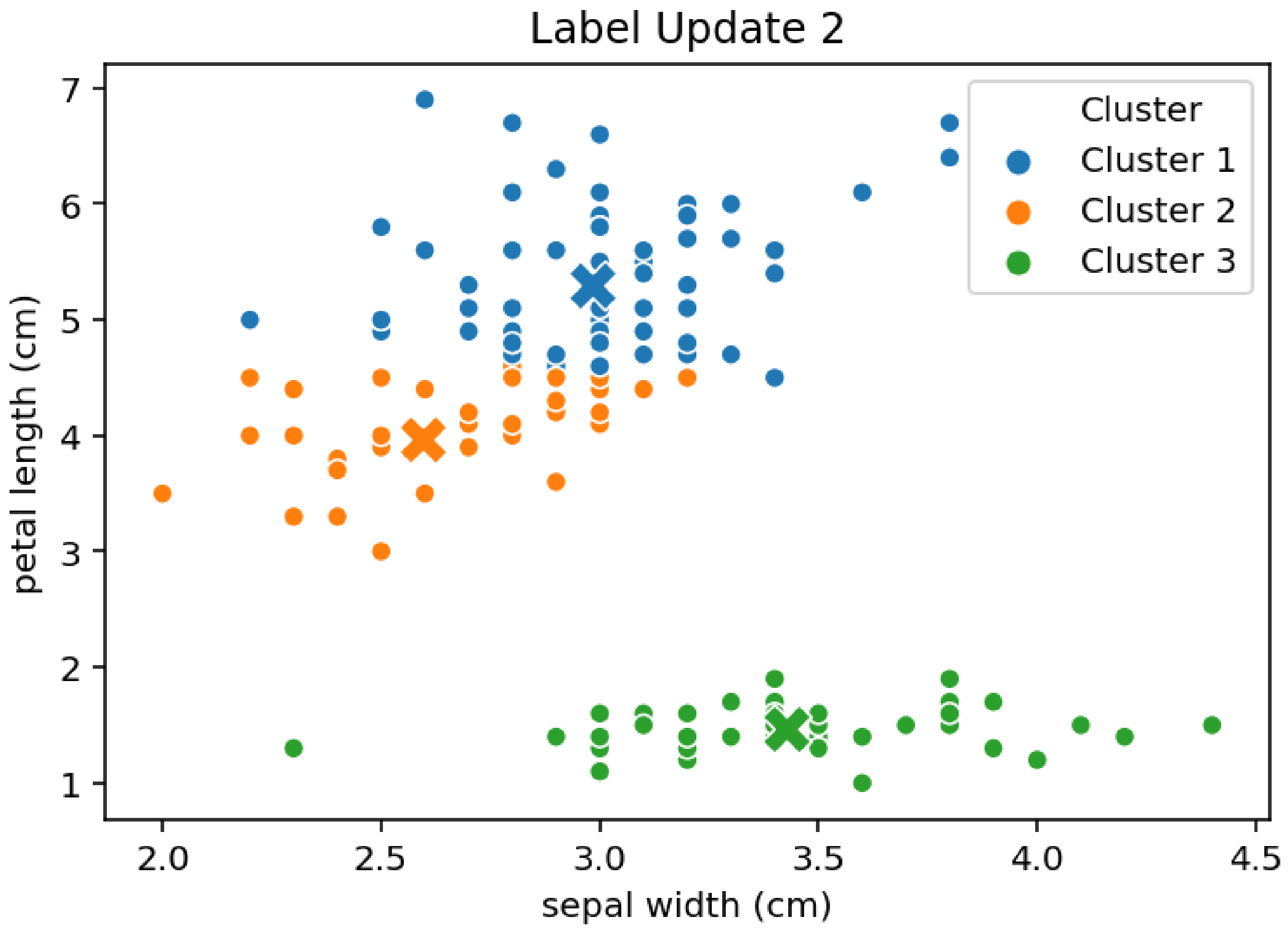

Centroid Update 4
Cluster
Cluster 1
Cluster 2
petal length (cm)
sepal width (cm)
7
6
5
4
3
2
1
2.0
2.5
3.0
3.5
4.0
4.5

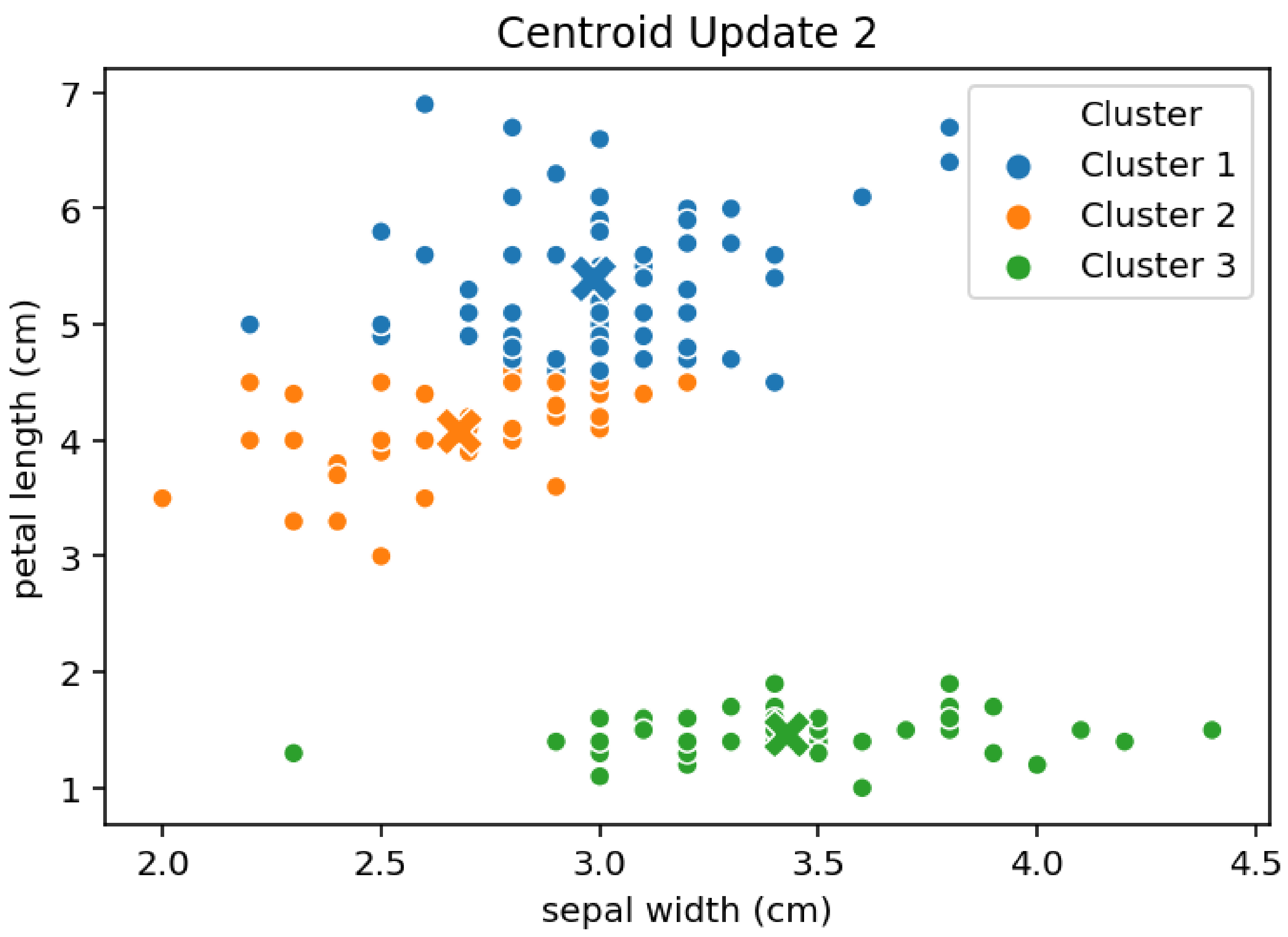

Label Update 5
Cluster
Cluster 1
Cluster 2
petal length (cm)
sepal width (cm)
1
2
3
4
5
6
7
2.0
2.5
3.0
3.5
4.0
4.5


Random Data Points Chosen As Centroids
petal length (cm)
sepal width (cm)
7
6
5
4
3
2
1
2.0
2.5
3.0
3.5
4.0
4.5
Centroid
Centroid 1
Centroid 2
Centroid 3

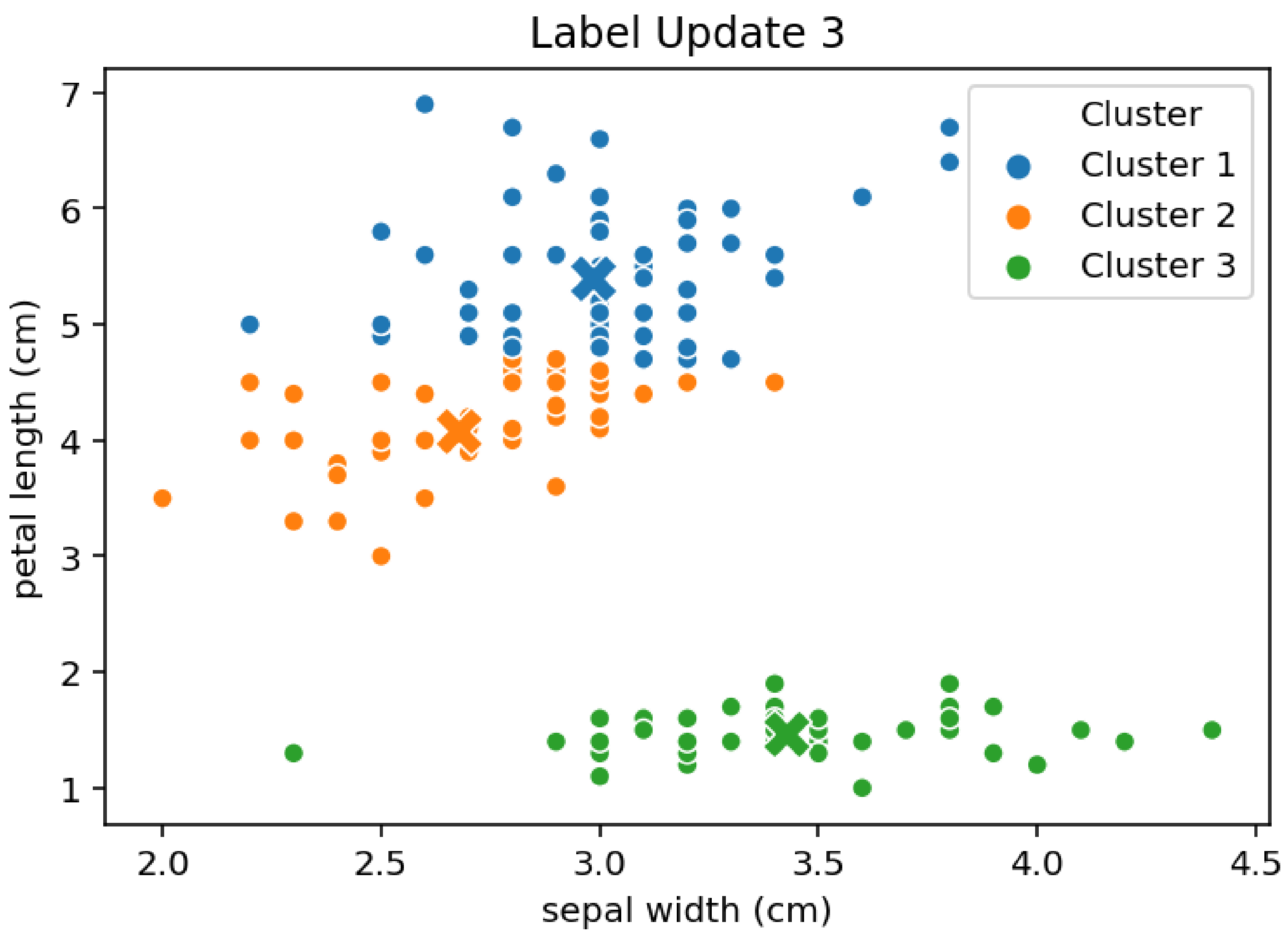

Label Update 1
Cluster
Cluster 1
Cluster 2
Cluster 3
petal length (cm)
sepal width (cm)
1
2
3
4
5
6
7
2.0
2.5
3.0
3.5
4.0
4.5

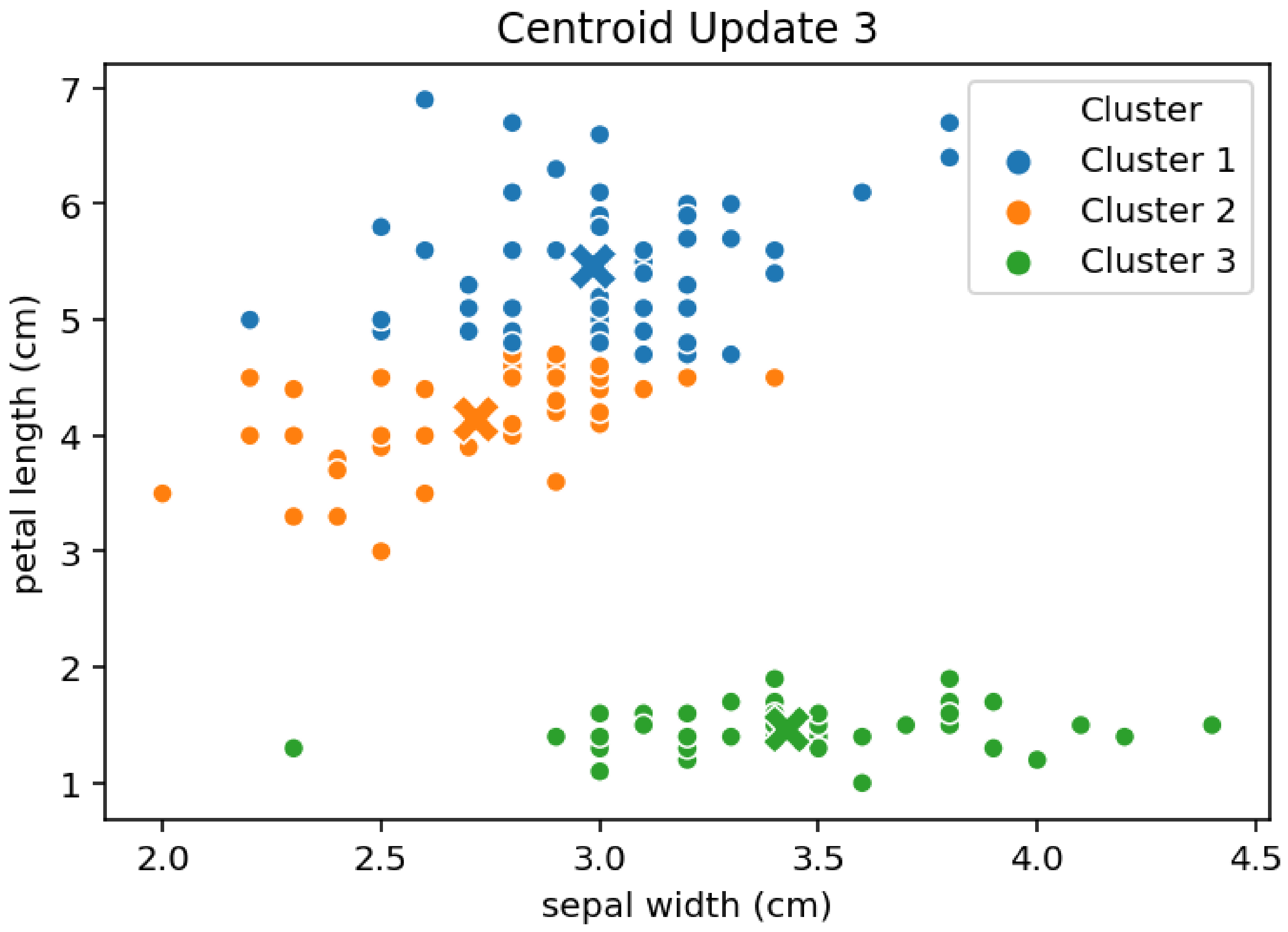

Centroid Update 1
Cluster
Cluster 1
Cluster 2
Cluster 3
petal length (cm)
sepal width (cm)
1
2
3
4
5
6
7
2.0
2.5
3.0
3.5
4.0
4.5

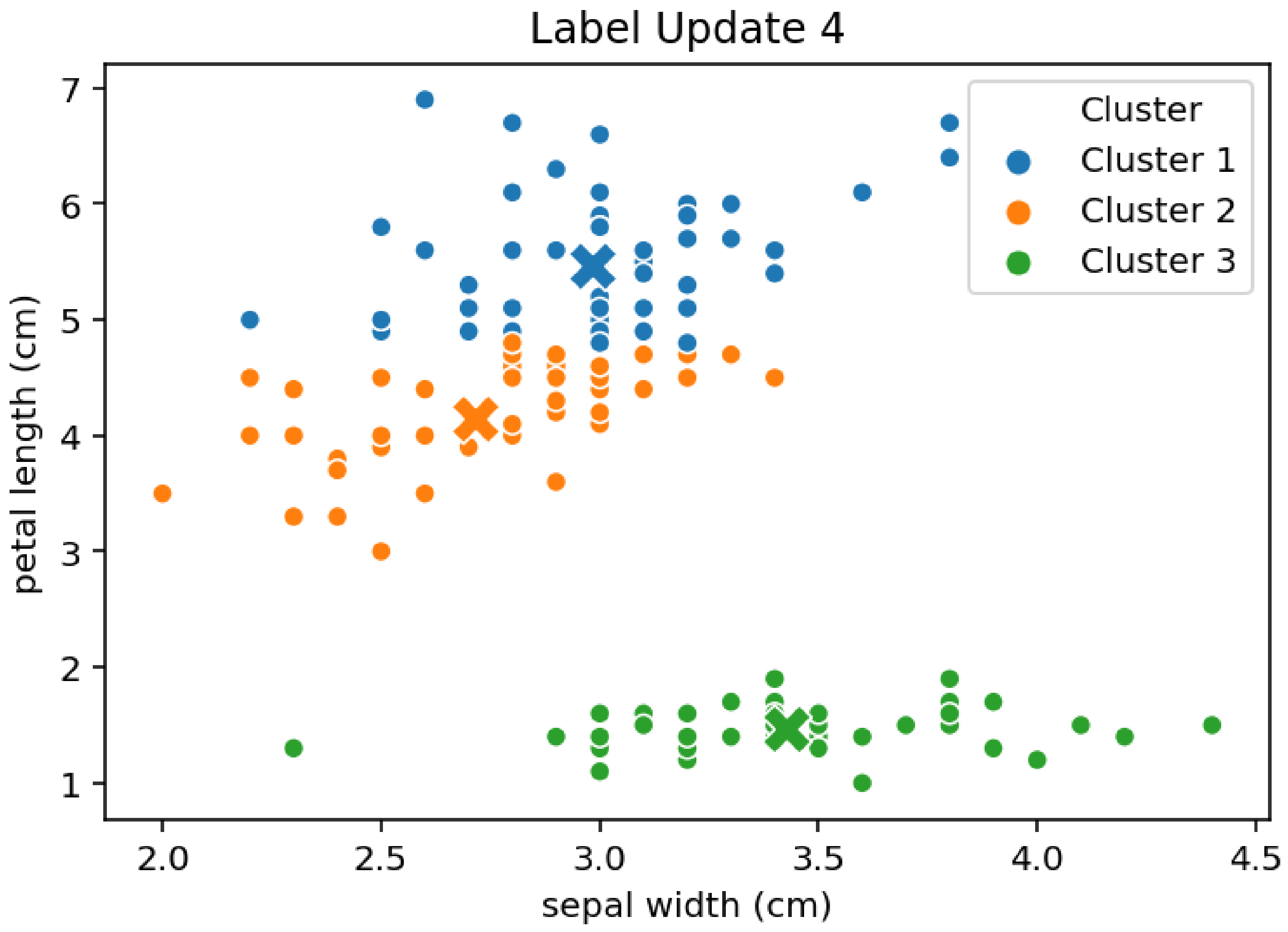

Label Update 2
Cluster
Cluster 1
Cluster 2
Cluster 3
petal length (cm)
sepal width (cm)
1
2
3
4
5
6
7
2.0
2.5
3.0
3.5
4.0
4.5

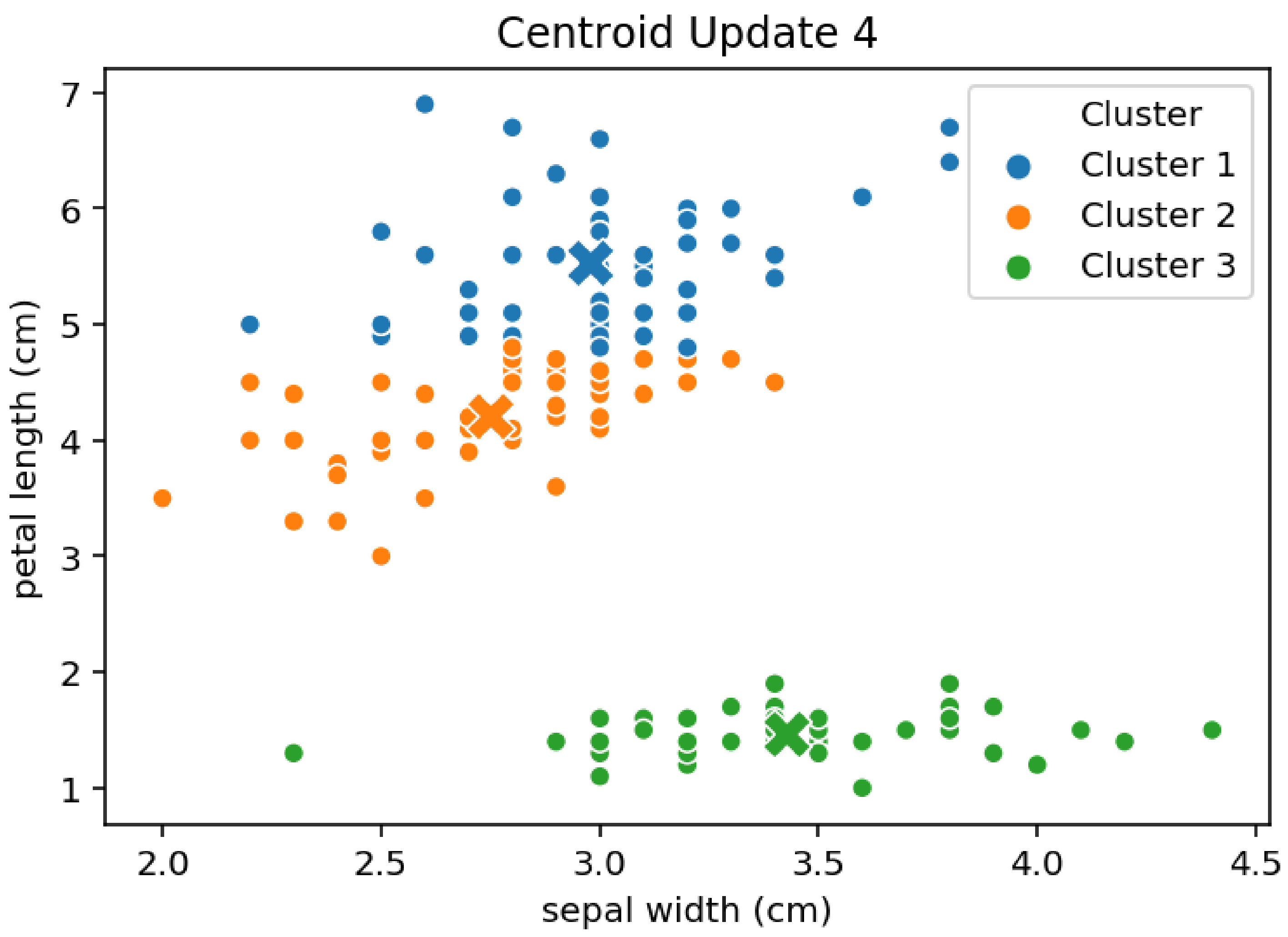


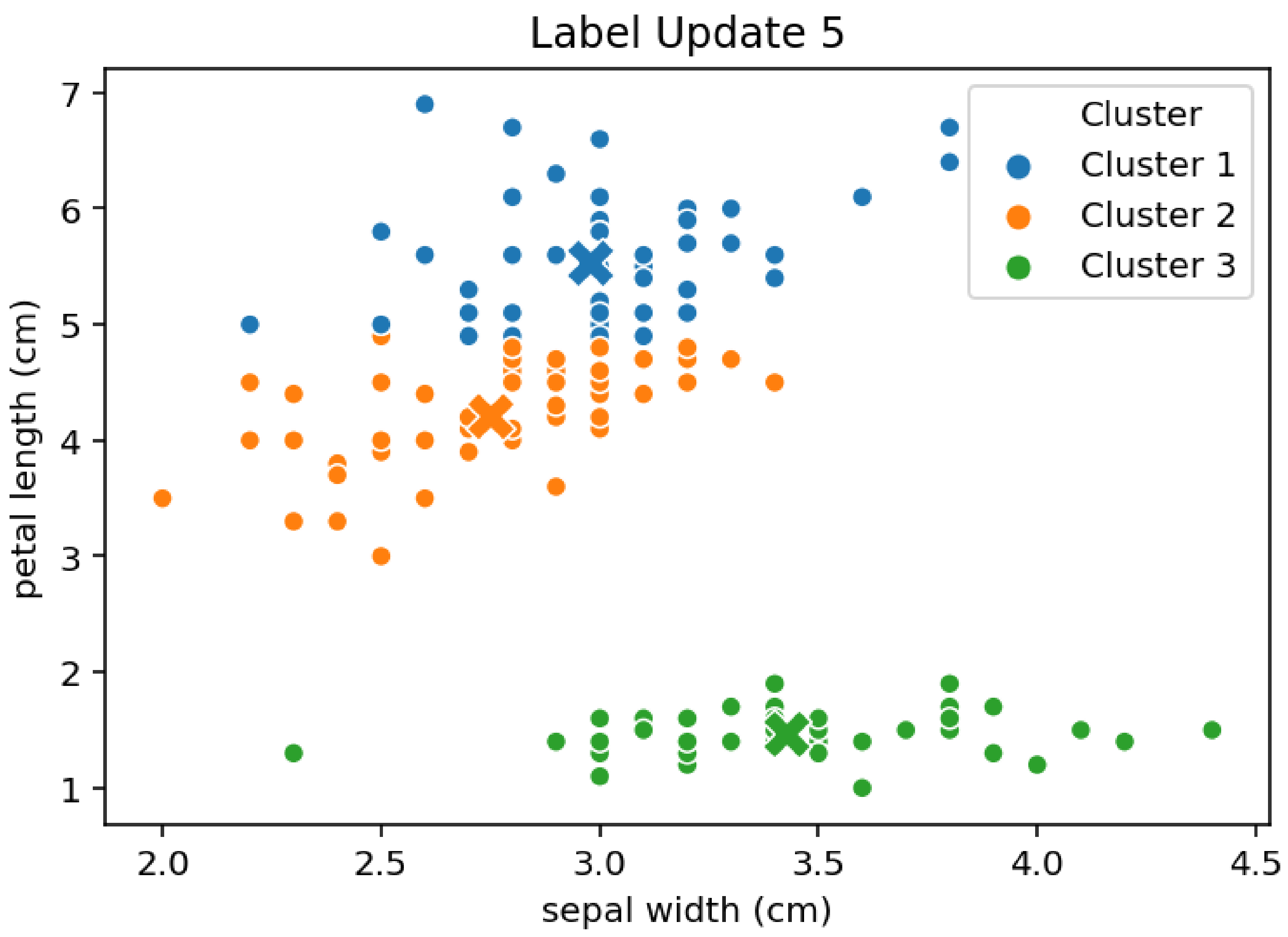
Centroid Update 2
Cluster
Cluster 1
Cluster 2
Cluster 3
petal length (cm)
sepal width (cm)
1
2
3
4
5
6
7
2.0
2.5
3.0
3.5
4.0
4.5

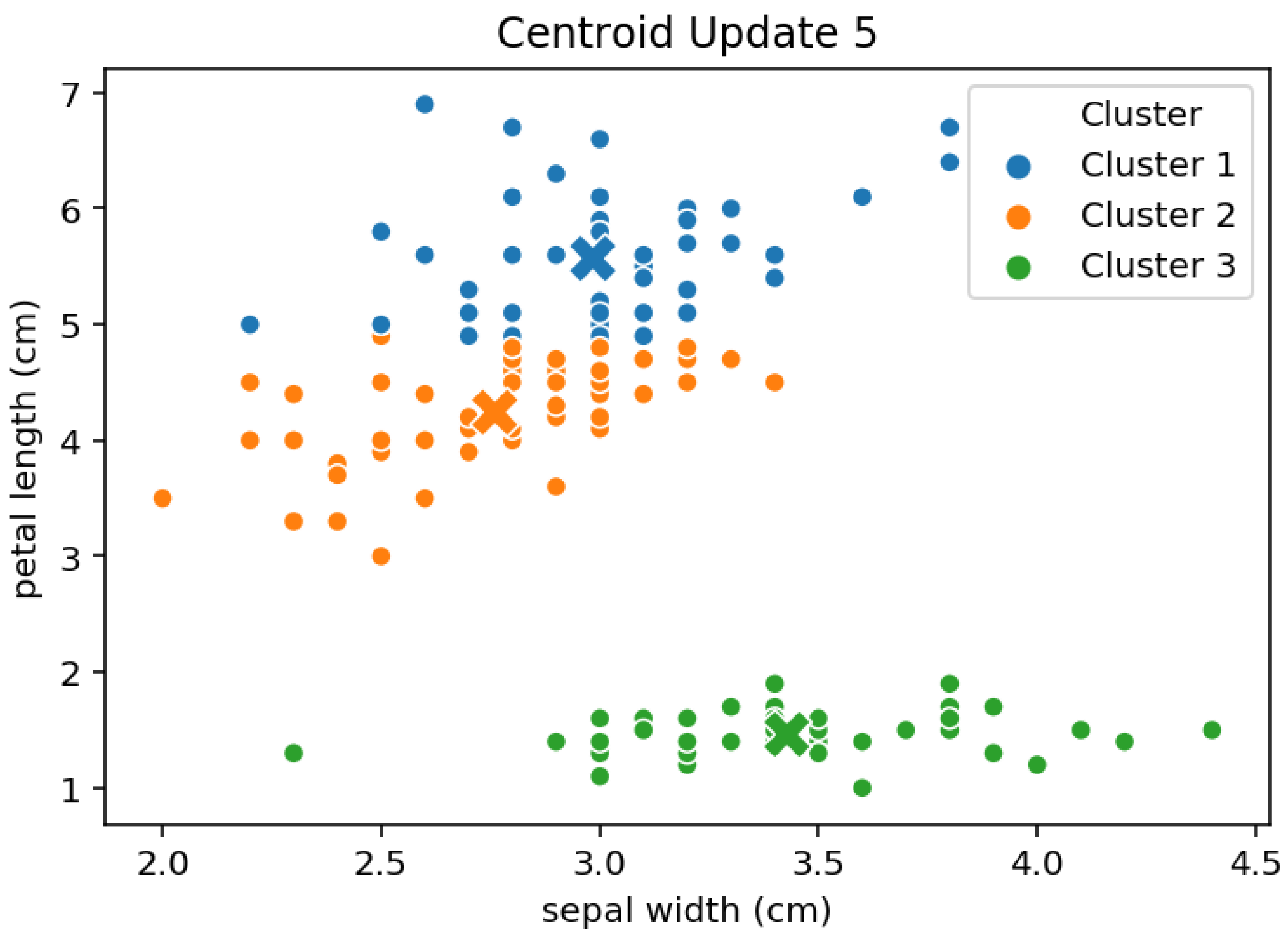

Label Update 3
Cluster
Cluster 1
Cluster 2
Cluster 3
petal length (cm)
sepal width (cm)
1
2
3
4
5
6
7
2.0
2.5
3.0
3.5
4.0
4.5

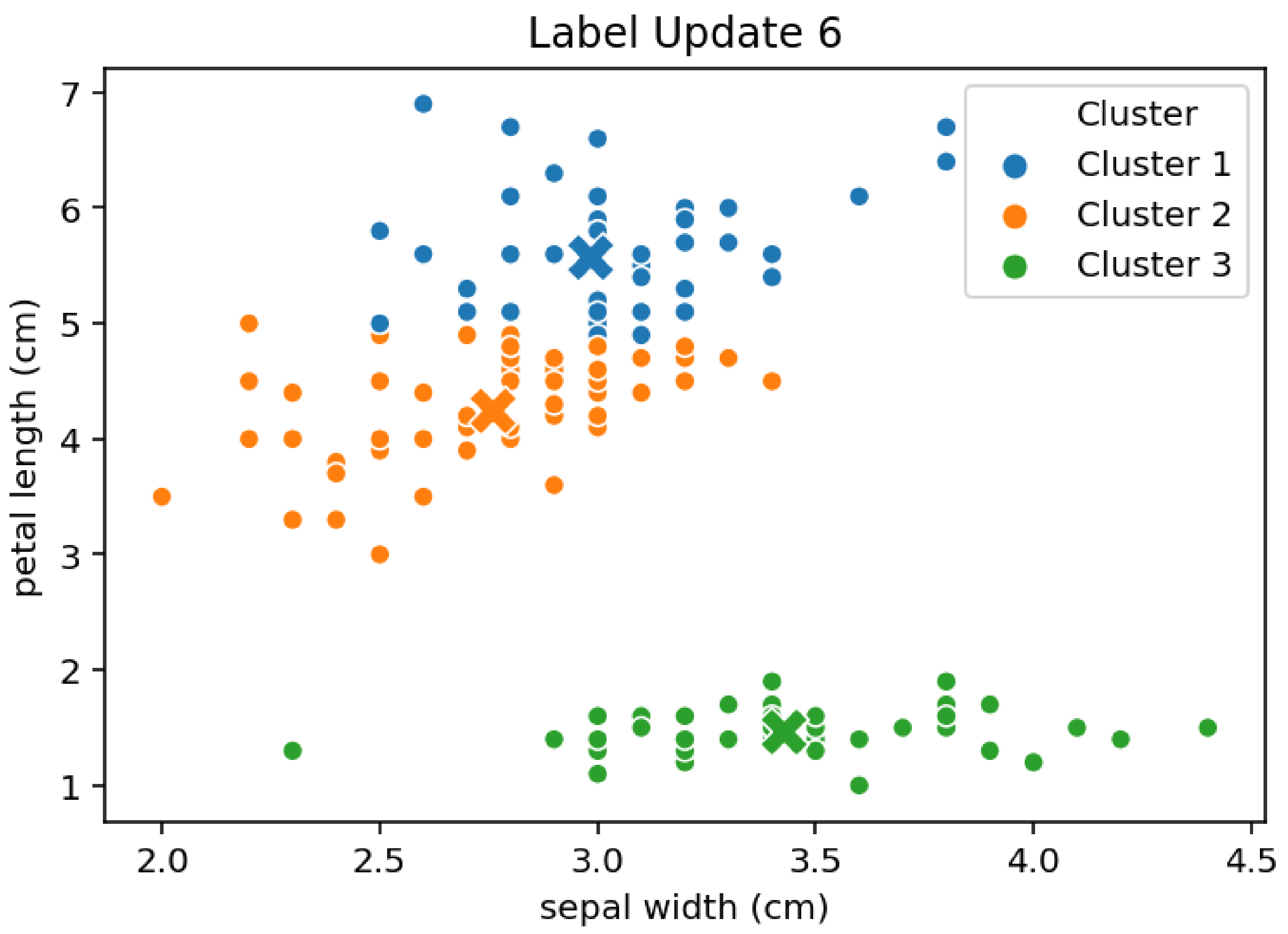

Centroid Update 3
petal length (cm)
sepal width (cm)
Cluster
Cluster 1
Cluster 2
Cluster 3
1
2
3
4
5
6
7
2.0
2.5
3.0
3.5
4.0
4.5

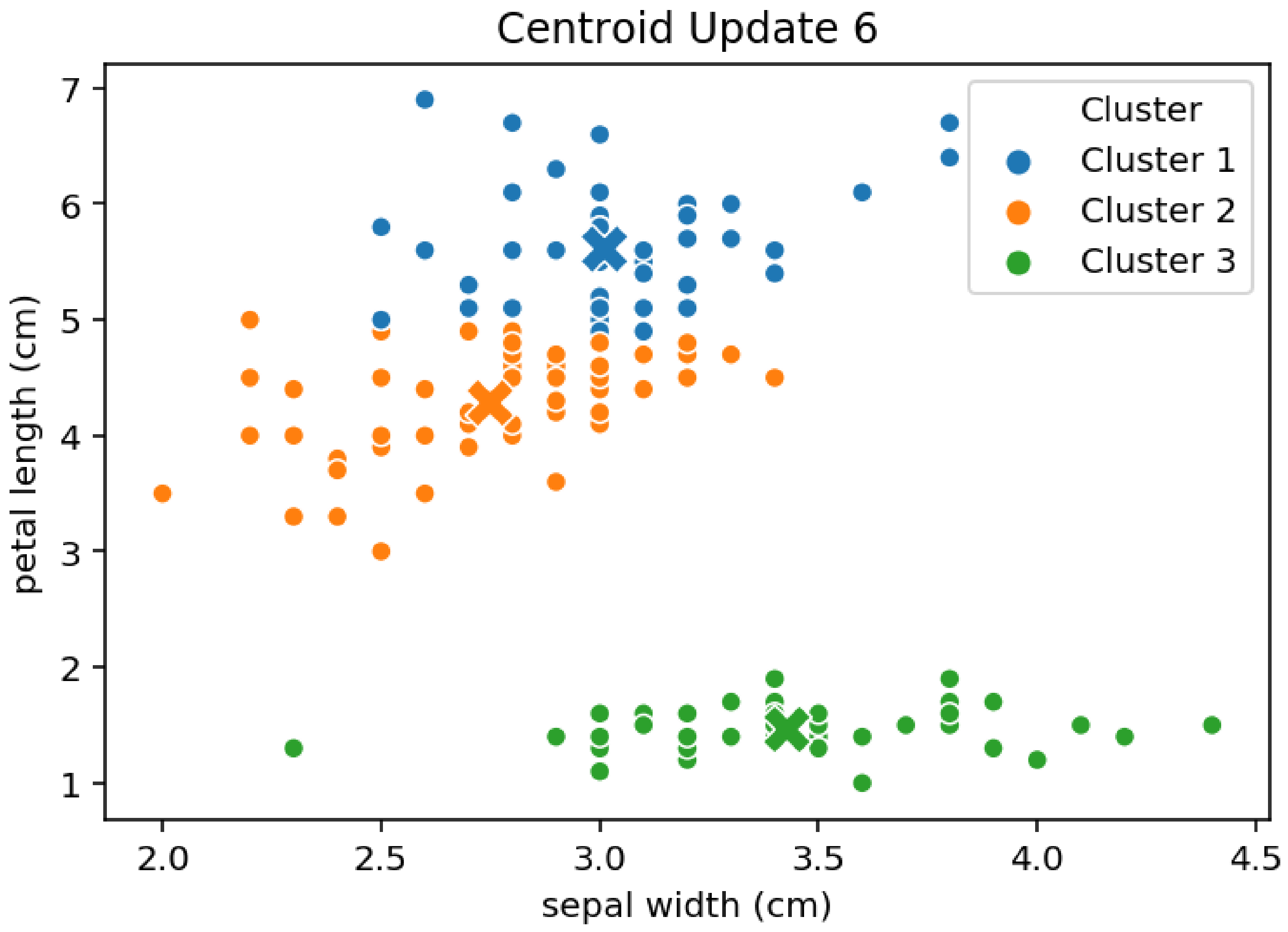

Label Update 4
petal length (cm)
sepal width (cm)
Cluster
Cluster 1
Cluster 2
Cluster 3
1
2
3
4
5
6
7
2.0
2.5
3.0
3.5
4.0
4.5

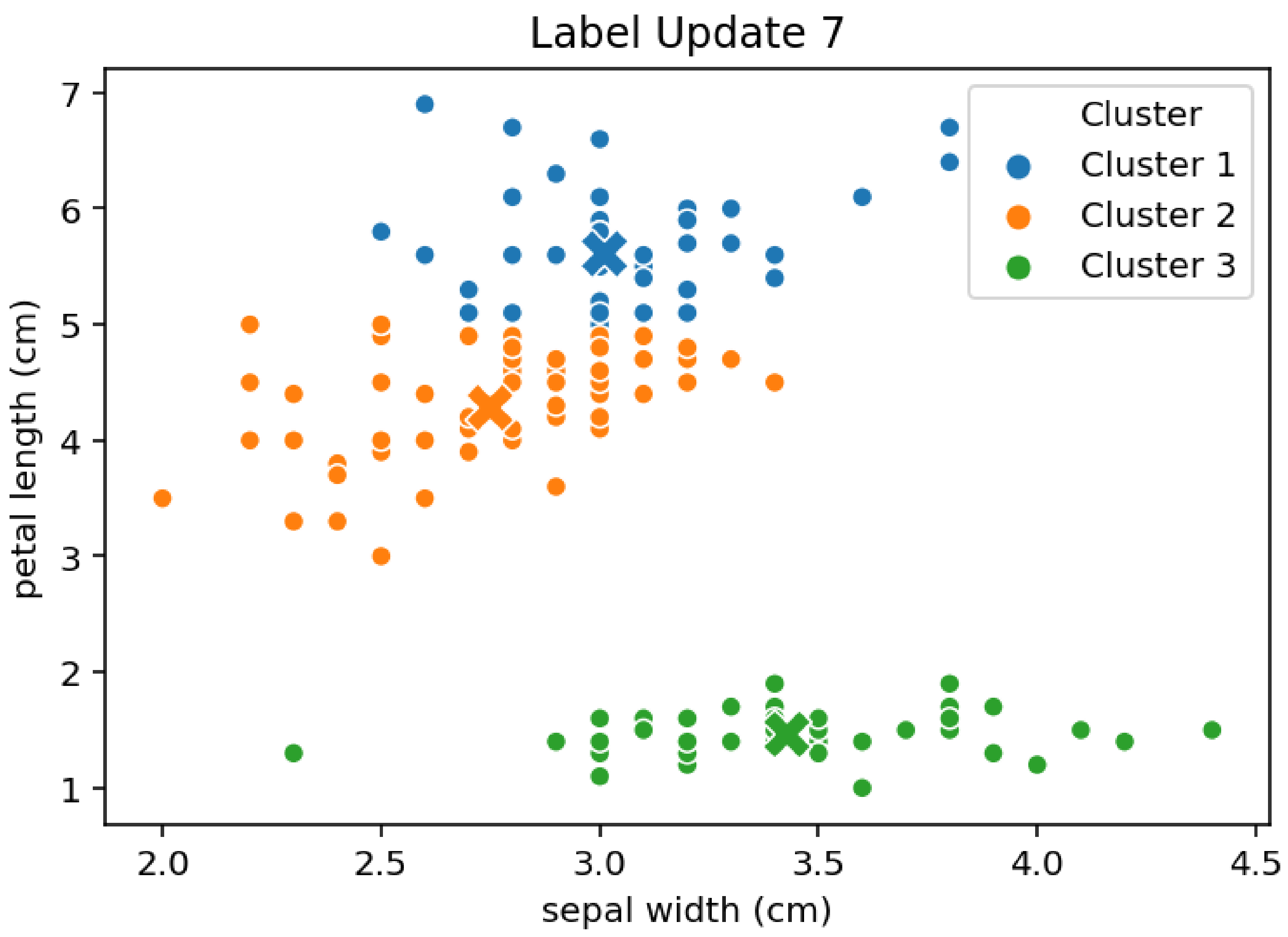

Centroid Update 4
petal length (cm)
sepal width (cm)
Cluster
Cluster 1
Cluster 2
Cluster 3
1
2
3
4
5
6
7
2.0
2.5
3.0
3.5
4.0
4.5

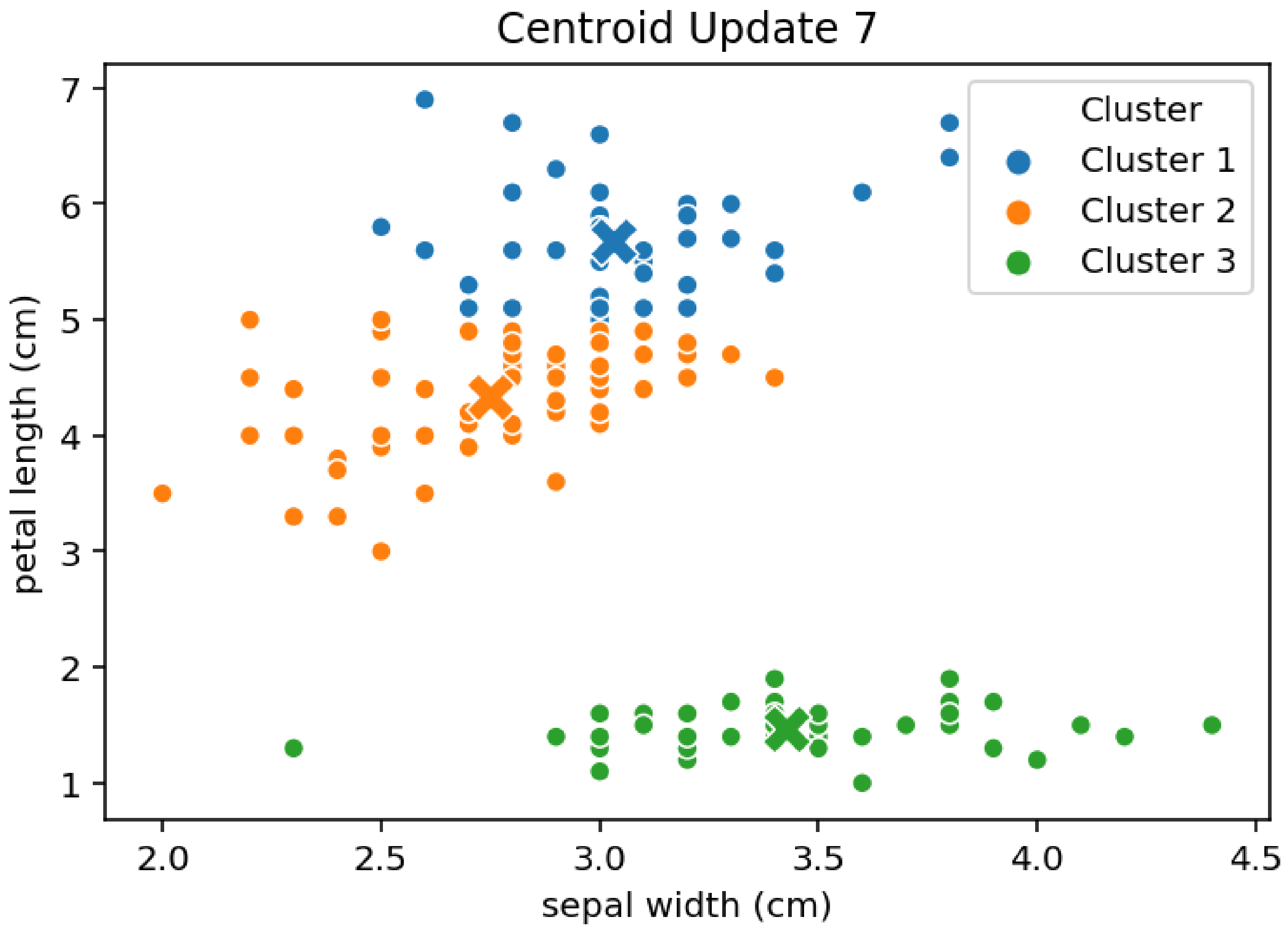

Label Update 5
Cluster
Cluster 1
Cluster 2
Cluster 3
petal length (cm)
sepal width (cm)
1
2
3
4
5
6
7
2.0
2.5
3.0
3.5
4.0
4.5

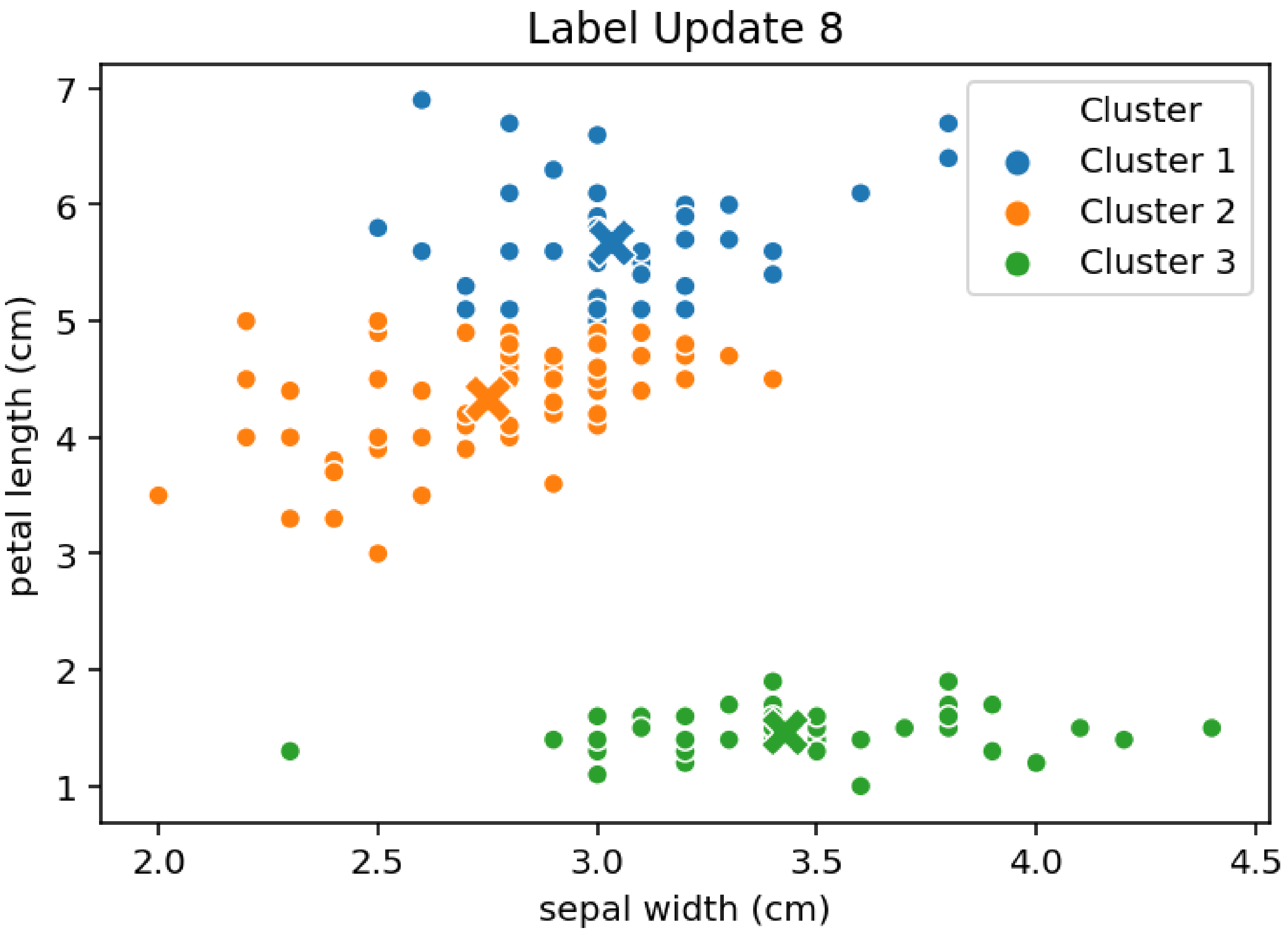

Centroid Update 5
petal length (cm)
sepal width (cm)
Cluster
Cluster 1
Cluster 2
Cluster 3
1
2
3
4
5
6
7
2.0
2.5
3.0
3.5
4.0
4.5


Label Update 6
Cluster
Cluster 1
Cluster 2
Cluster 3
petal length (cm)
sepal width (cm)
1
2
3
4
5
6
7
2.0
2.5
3.0
3.5
4.0
4.5


Centroid Update 6
Cluster
Cluster 1
Cluster 2
Cluster 3
petal length (cm)
sepal width (cm)
7
6
5
4
3
2
1
2.0
2.5
3.0
3.5
4.0
4.5


Label Update 7
Cluster
Cluster 1
Cluster 2
Cluster 3
petal length (cm)
sepal width (cm)
1
2
3
4
5
6
7
2.0
2.5
3.0
3.5
4.0
4.5


Centroid Update 7
petal length (cm)
sepal width (cm)
Cluster
Cluster 1
Cluster 2
Cluster 3
1
2
3
4
5
6
7
2.0
2.5
3.0
3.5
4.0
4.5

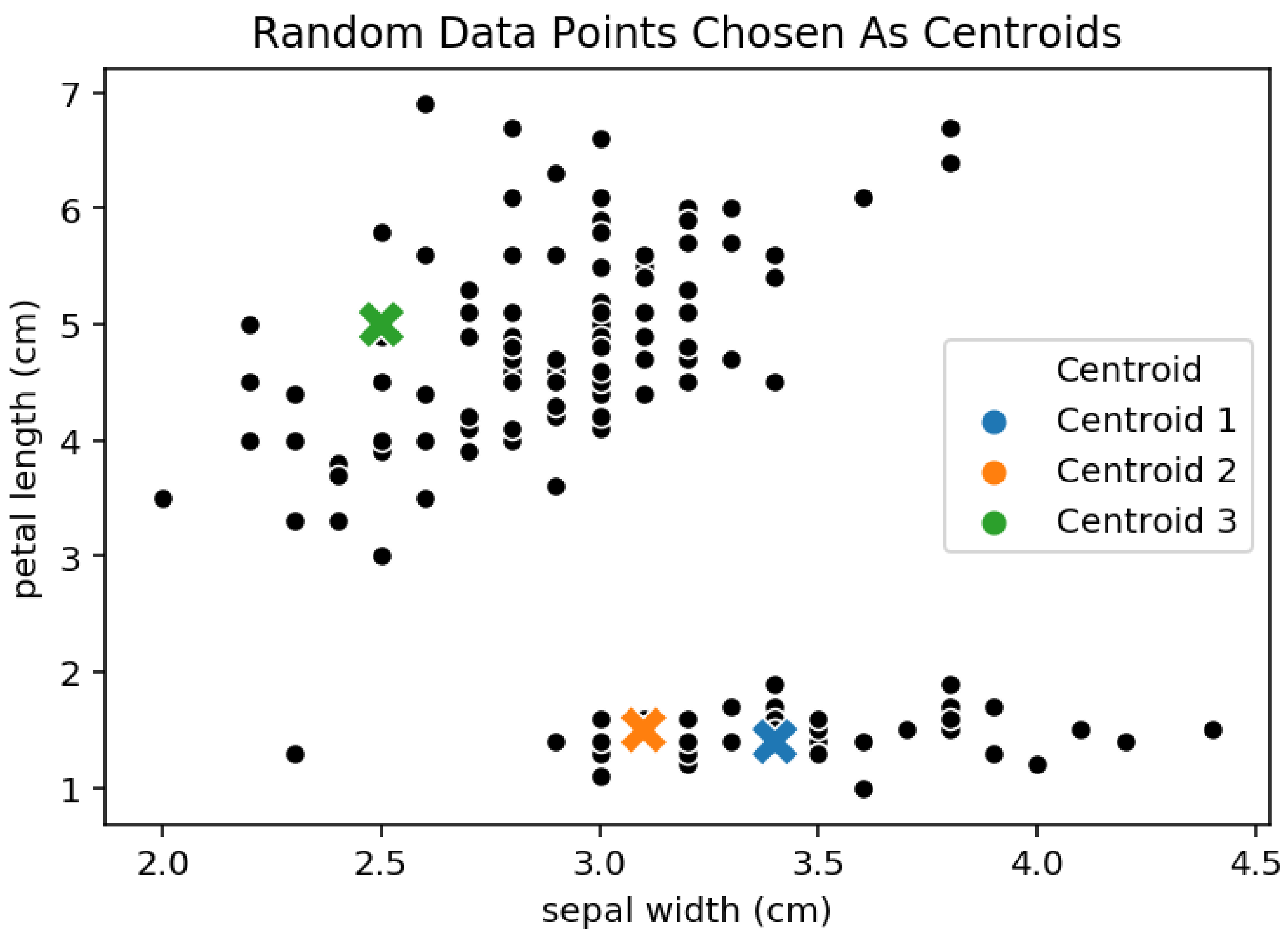

Label Update 8
Cluster
Cluster 1
Cluster 2
Cluster 3
petal length (cm)
sepal width (cm)
7
6
5
4
3
2
1
2.0
2.5
3.0
3.5
4.0
4.5

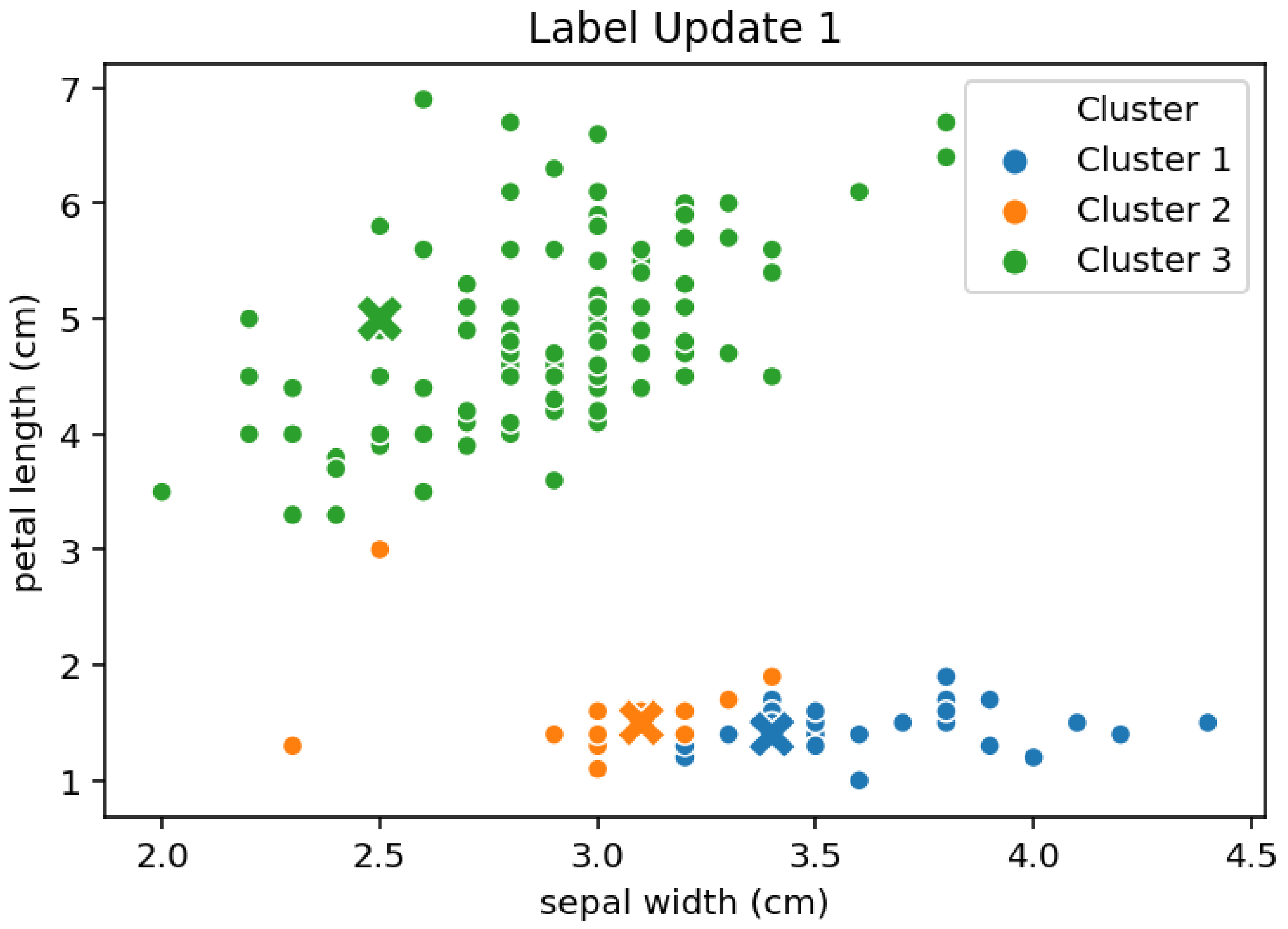
Questions

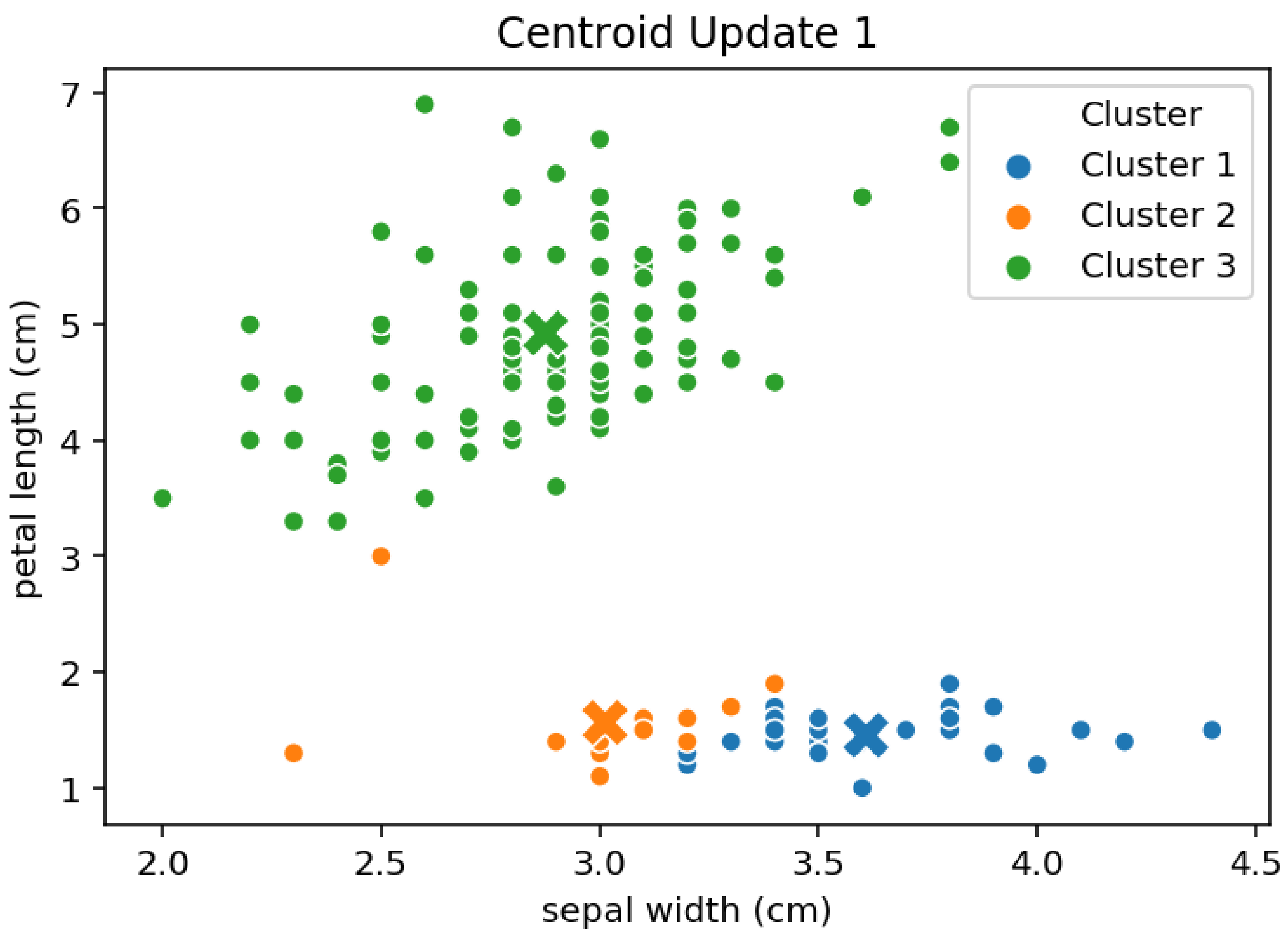
Will K-means terminate?

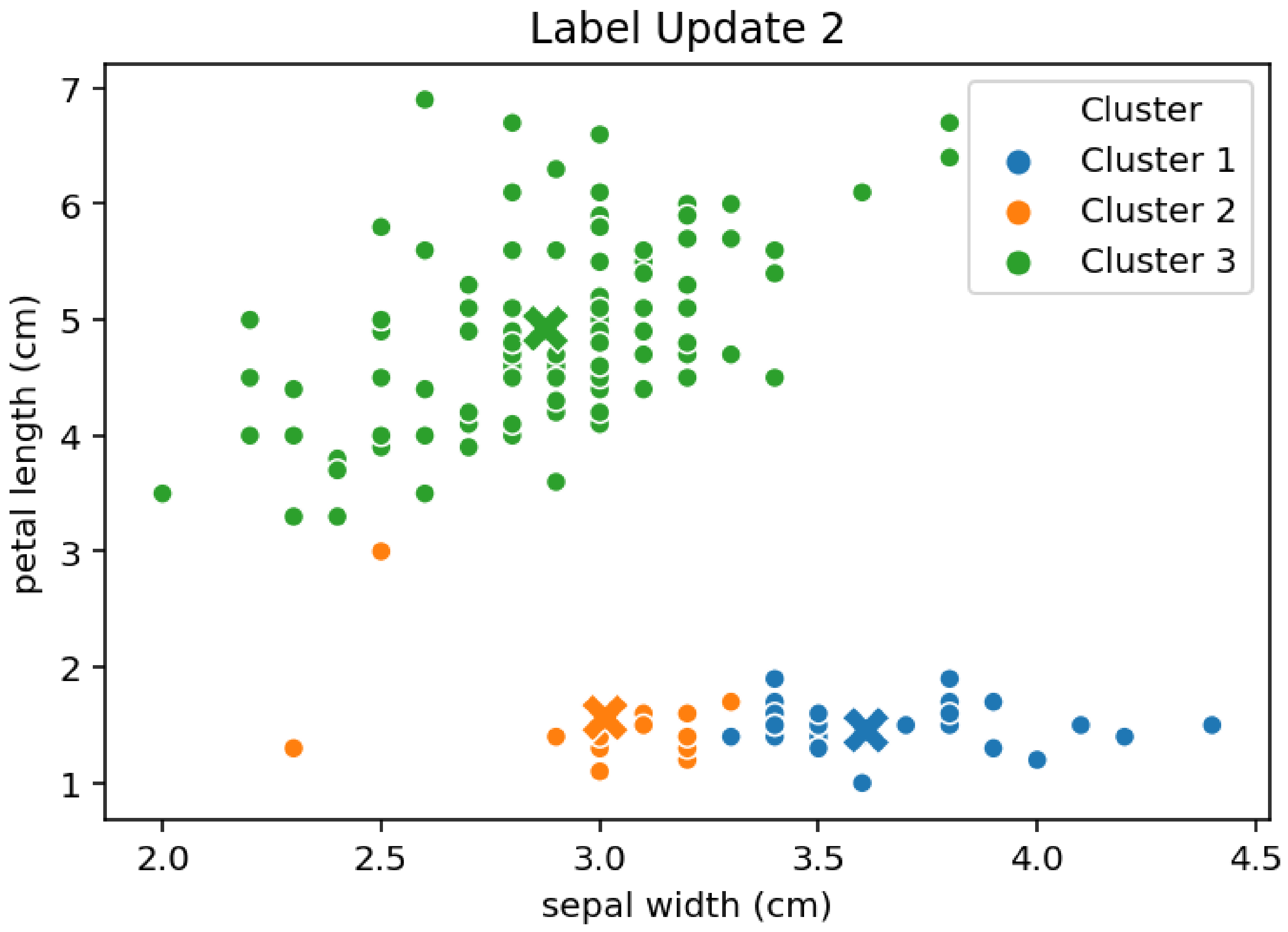
Will it always find the same answer?

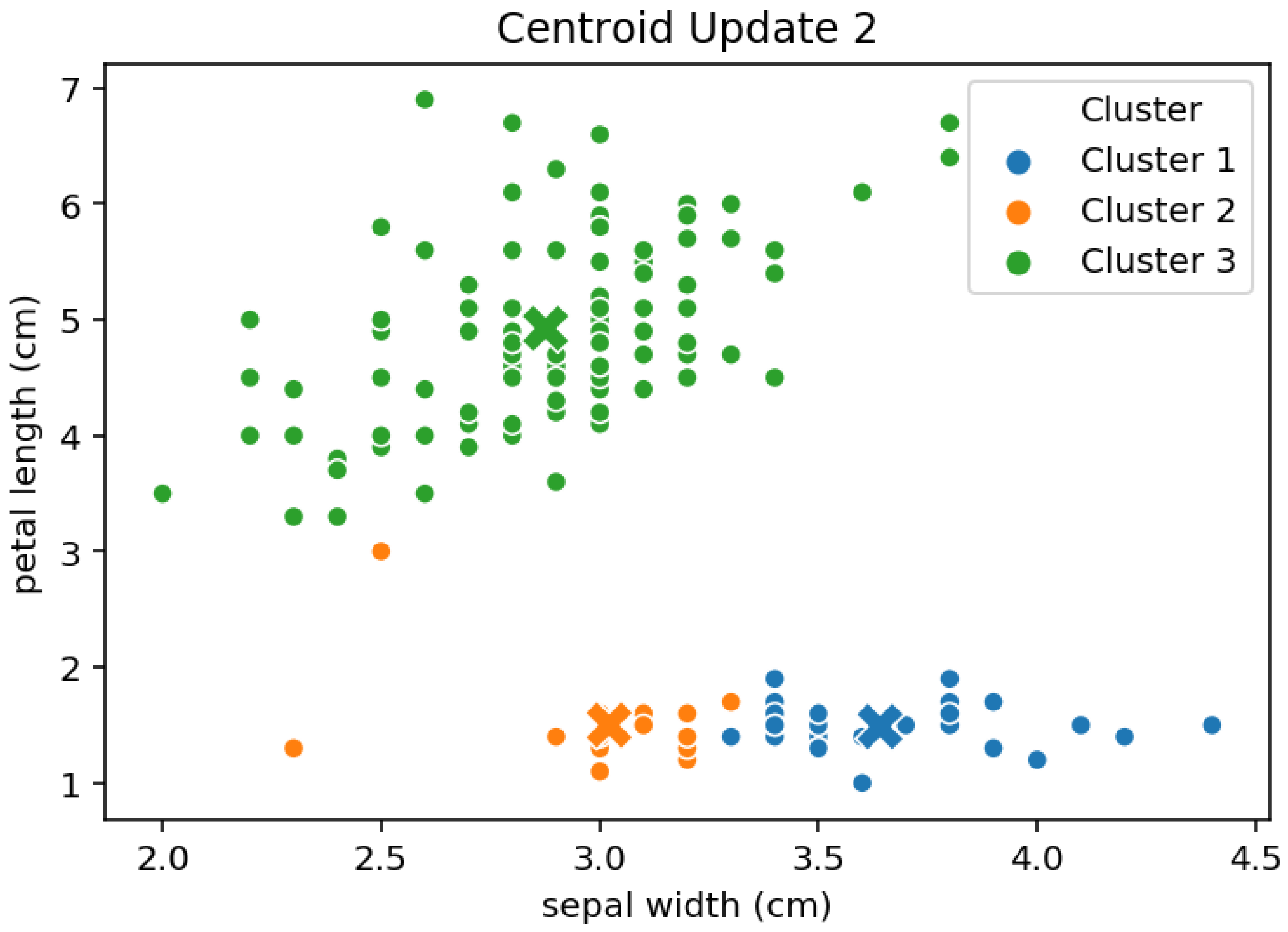
How should we choose the number of clusters?

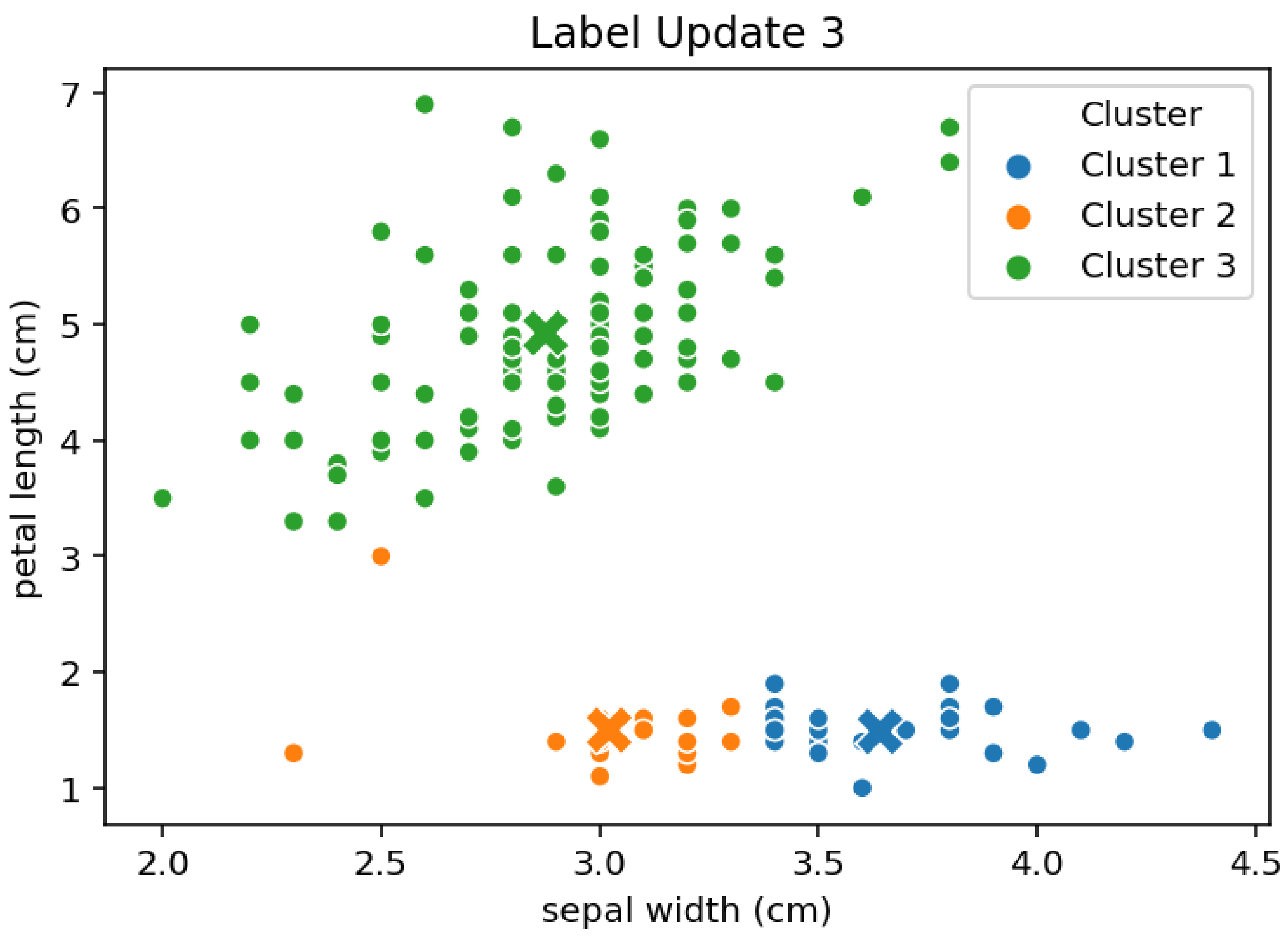

Random Data Points Chosen As Centroids
petal length (cm)
sepal width (cm)
Centroid
Centroid 1
Centroid 2
Centroid 3

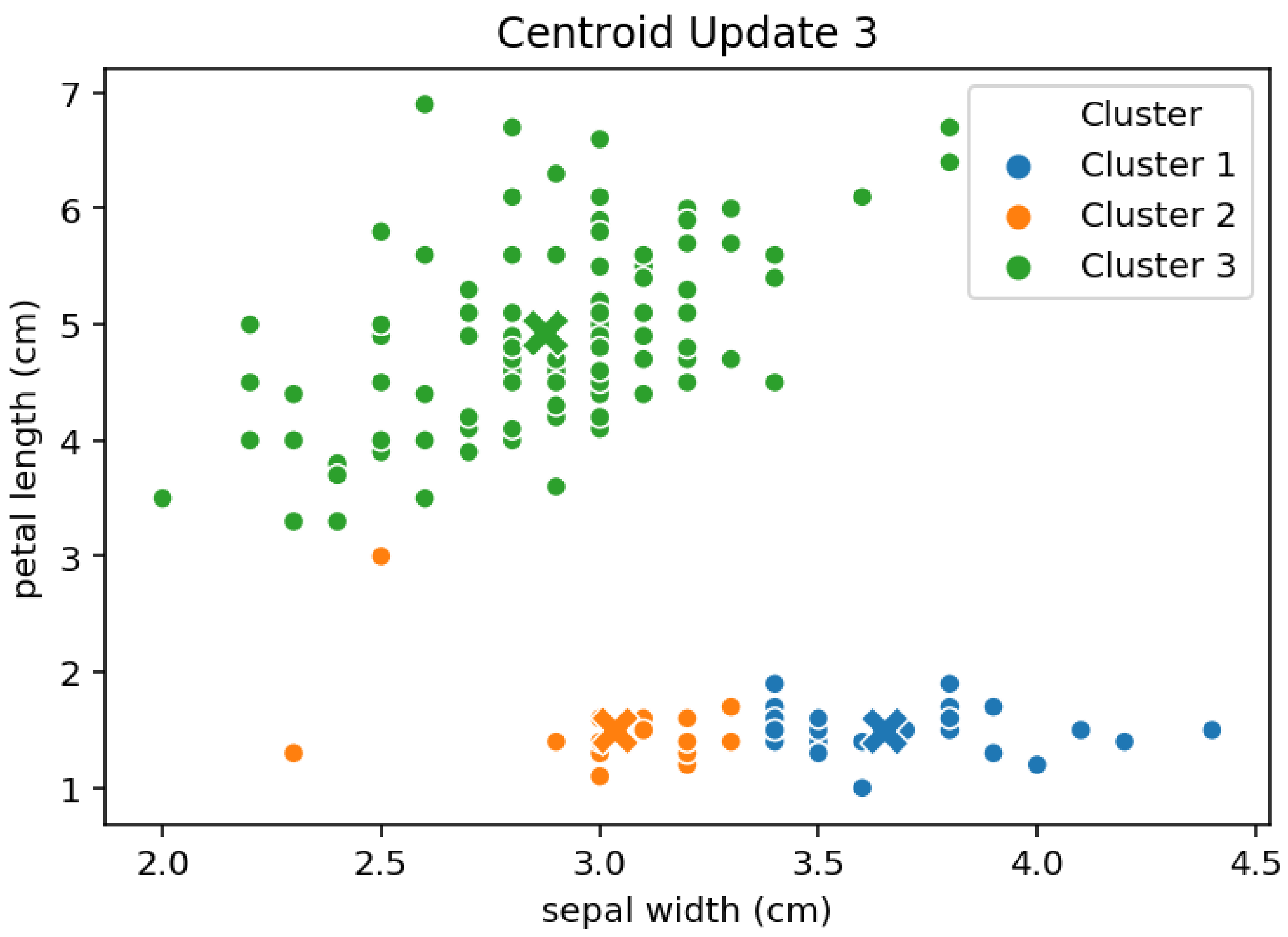

Label Update 1
Cluster
Cluster 1
Cluster 2
Cluster 3
petal length (cm)
sepal width (cm)
1
2
3
4
5
6
7
2.0
2.5
3.0
3.5
4.0
4.5

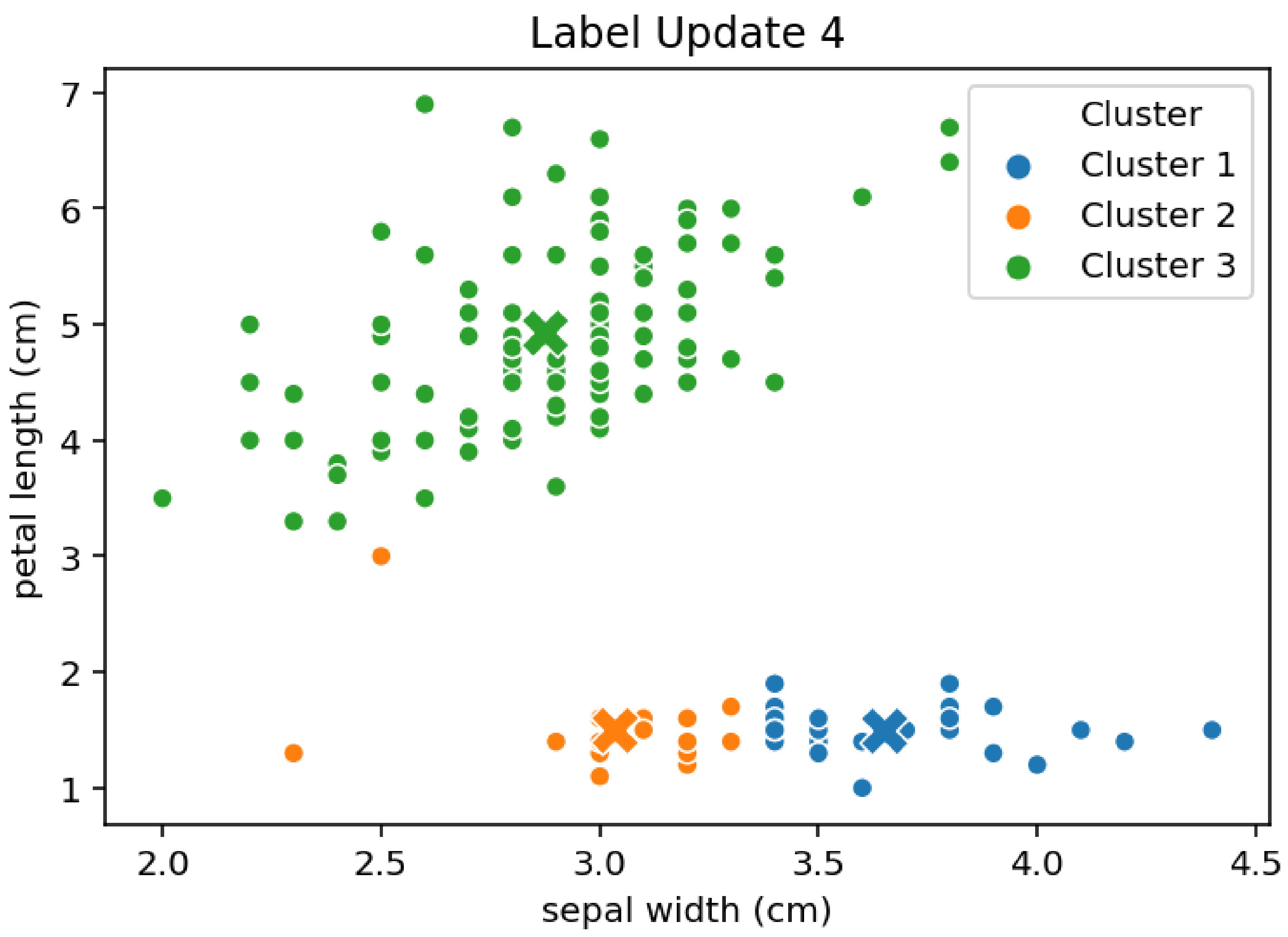

Centroid Update 1
Cluster
Cluster 1
Cluster 2
Cluster 3
petal length (cm)
sepal width (cm)
1
2
3
4
5
6
7
2.0
2.5
3.0
3.5
4.0
4.5


Label Update 2
Cluster
Cluster 1
Cluster 2
Cluster 3
petal length (cm)
sepal width (cm)
1
2
3
4
5
6
7
2.0
2.5
3.0
3.5
4.0
4.5


Centroid Update 2
Cluster
Cluster 1
Cluster 2
Cluster 3
petal length (cm)
sepal width (cm)
1
2
3
4
5
6
7
2.0
2.5
3.0
3.5
4.0
4.5


Label Update 3
Cluster
Cluster 1
Cluster 2
Cluster 3
petal length (cm)
sepal width (cm)
1
2
3
4
5
6
7
2.0
2.5
3.0
3.5
4.0
4.5


Centroid Update 3
Cluster
Cluster 1
Cluster 2
Cluster 3
petal length (cm)
sepal width (cm)
1
2
3
4
5
6
7
2.0
2.5
3.0
3.5
4.0
4.5


Label Update 4
petal length (cm)
sepal width (cm)
Cluster
Cluster 1
Cluster 2
Cluster 3

Will K-means always find the same answer?

K-means is a version of coordinate descent, where the parameters are the cluster center coordinates, and the assignments of points to clusters.

This error function has many local minima!

The solution found is *locally optimal*, but *not globally optimal*

Because the solution depends on the initial assignment of instances to clusters, random restarts will give different solutions

- Algorithms fix this by restarting many times! See the lab.

Choosing the Number of Clusters

How should we choose the number of clusters?

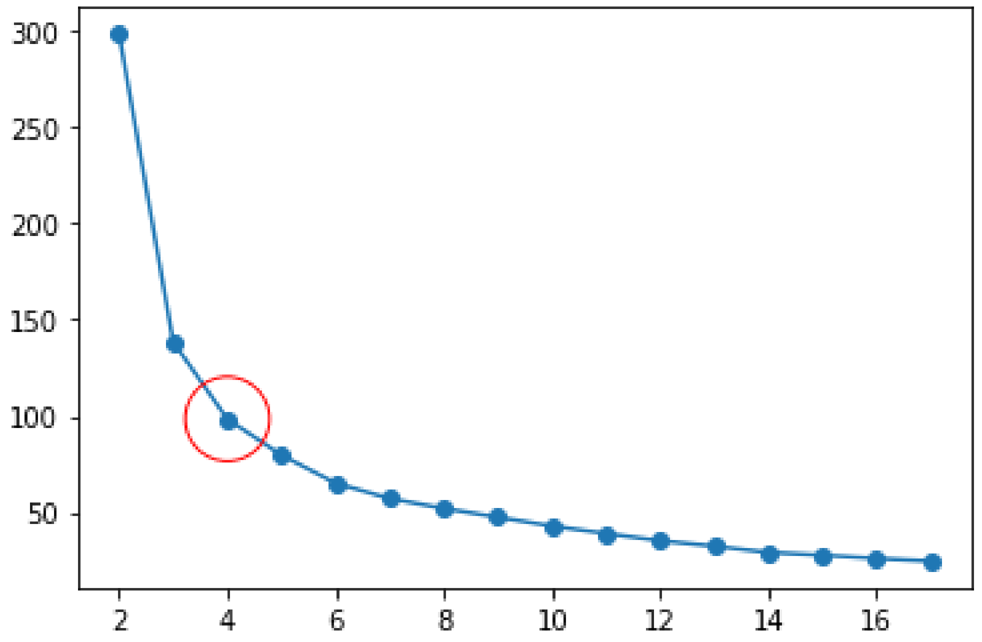
K-means criterion

$$\sum_{i=1}^{K} \sum_{j \in C_i} \| \mathbf{x}_j - \mathbf{c}_i \|^2$$

Suppose I have a clustering with *k* clusters. What can happen to this loss if I use an additional cluster?

Elbow

"Elbow" a.k.a. "stop adding clusters when it doesn't help much."



This person liked 4 clusters. Why not 5?

Formalizing the Elbow Method

There is a way to formally define the elbow!

- An elbow exists when the curve has **maximum curvature**.
- Read Satopää et al (2011). https://raghavan.usc.edu//papers/kneedle-simplex11.pdf

For any continuous function f exists another function K_f (the curvature) having the form:

$$K_f(x) = \frac{f''(x)}{(1 + f'(x)^2)^{1.5}}$$

The problem of finding the knee is done by maximizing this curve over discrete datasets (a bit of a more complex problem) that is implemented in the kneed package (https://github.com/arvkevi/kneed)

Silhouette

“Silhouette” score measures how close a point is to other points in its own cluster, compared with the “nearest” other cluster.

- Range [-1,1]. 1 means much closer to own cluster, 0 means equally close to own cluster as to another cluster. Negative means closer to another cluster.

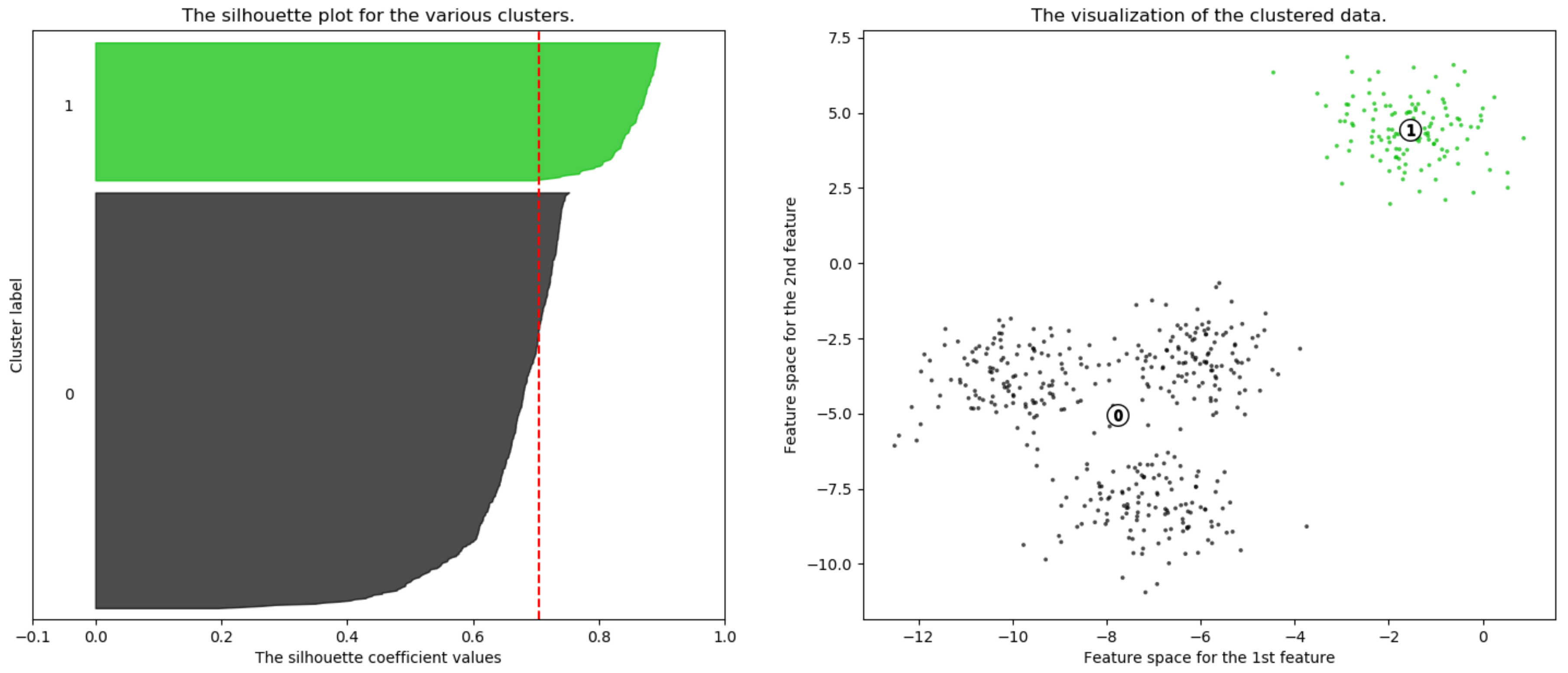
- $a(i) = \frac{1}{|C_i|-1} \sum_{j \in C_i, i \neq j} d(i,j), \quad b(i) = \min_{k \neq i} \frac{1}{|C_k|} \sum_{j \in C_k} d(i,j)$

- $s(i) = \frac{b(i)-a(i)}{max\{a(i),b(i)\}}$

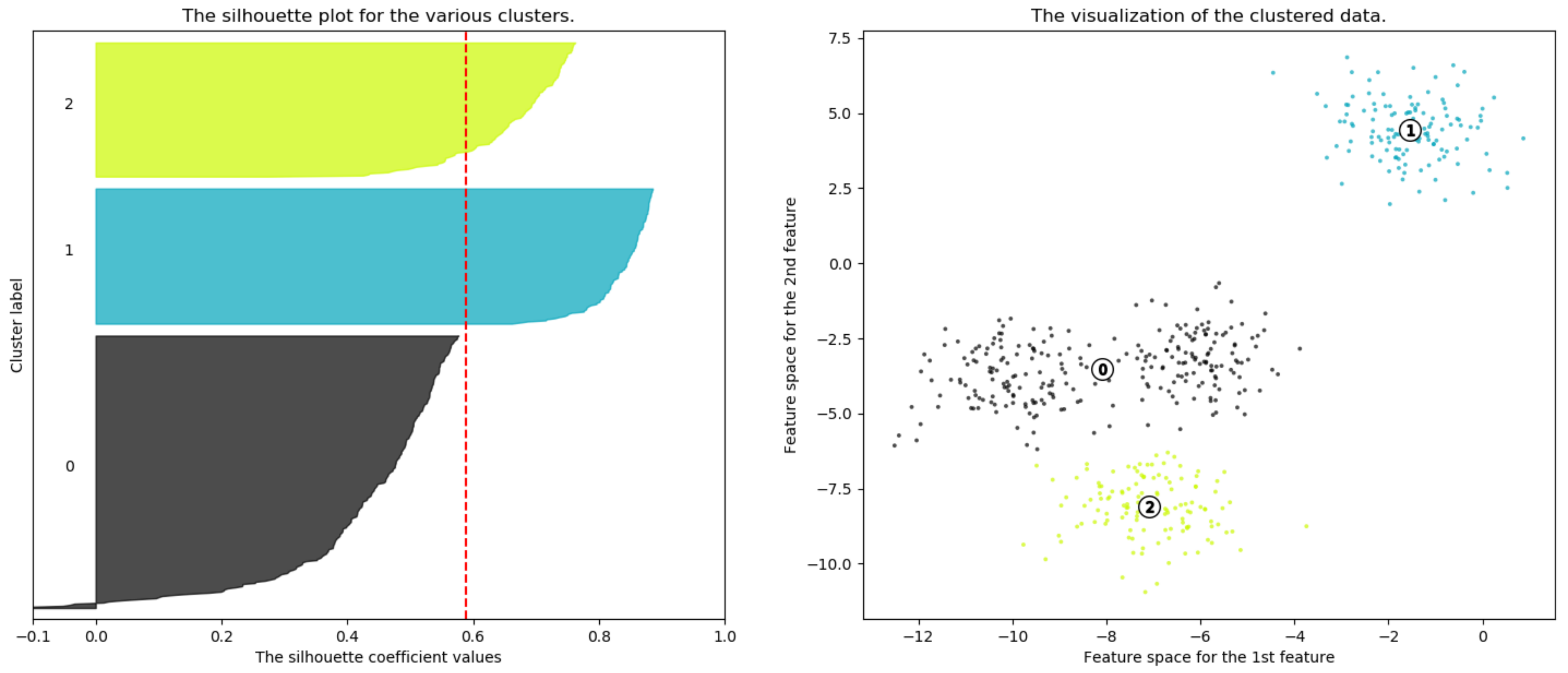
- Clusters with 1 point have silhouette 0 by definition.

https://scikit-learn.org/stable/auto_examples/cluster/plot_kmeans_silhouette_analysis.html

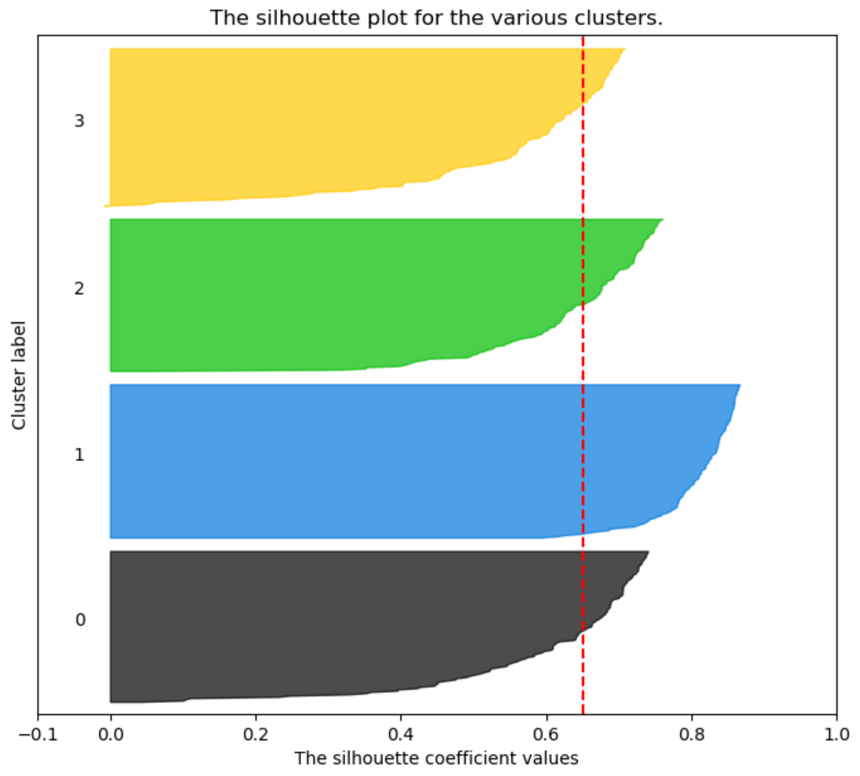
Silhouette analysis for KMeans clustering on sample data with n_clusters = 2

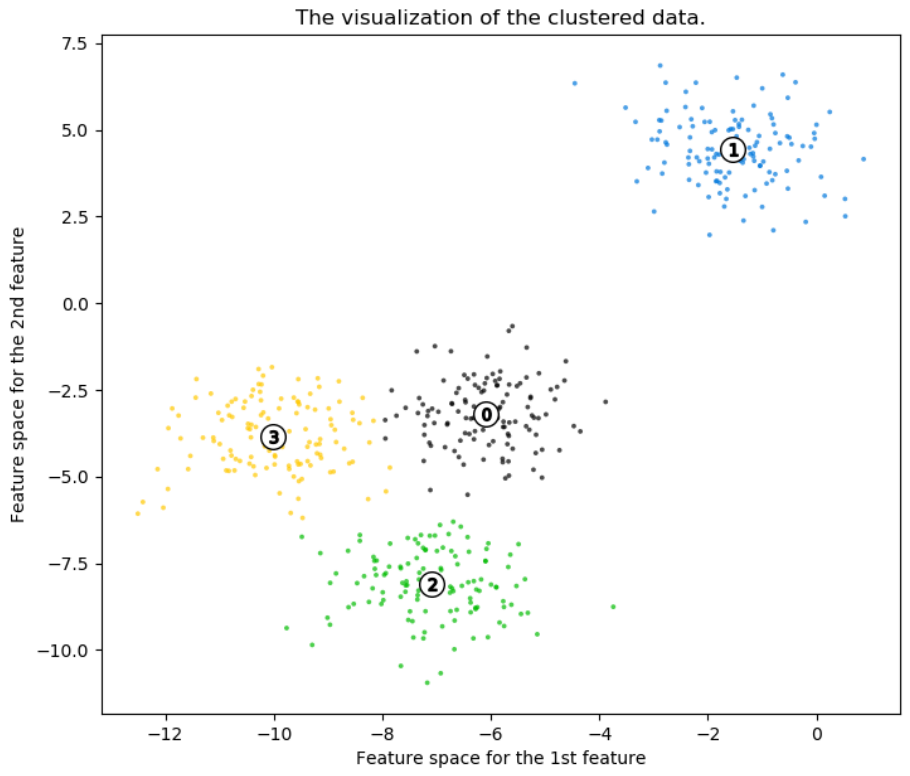


Silhouette analysis for KMeans clustering on sample data with n_clusters = 3

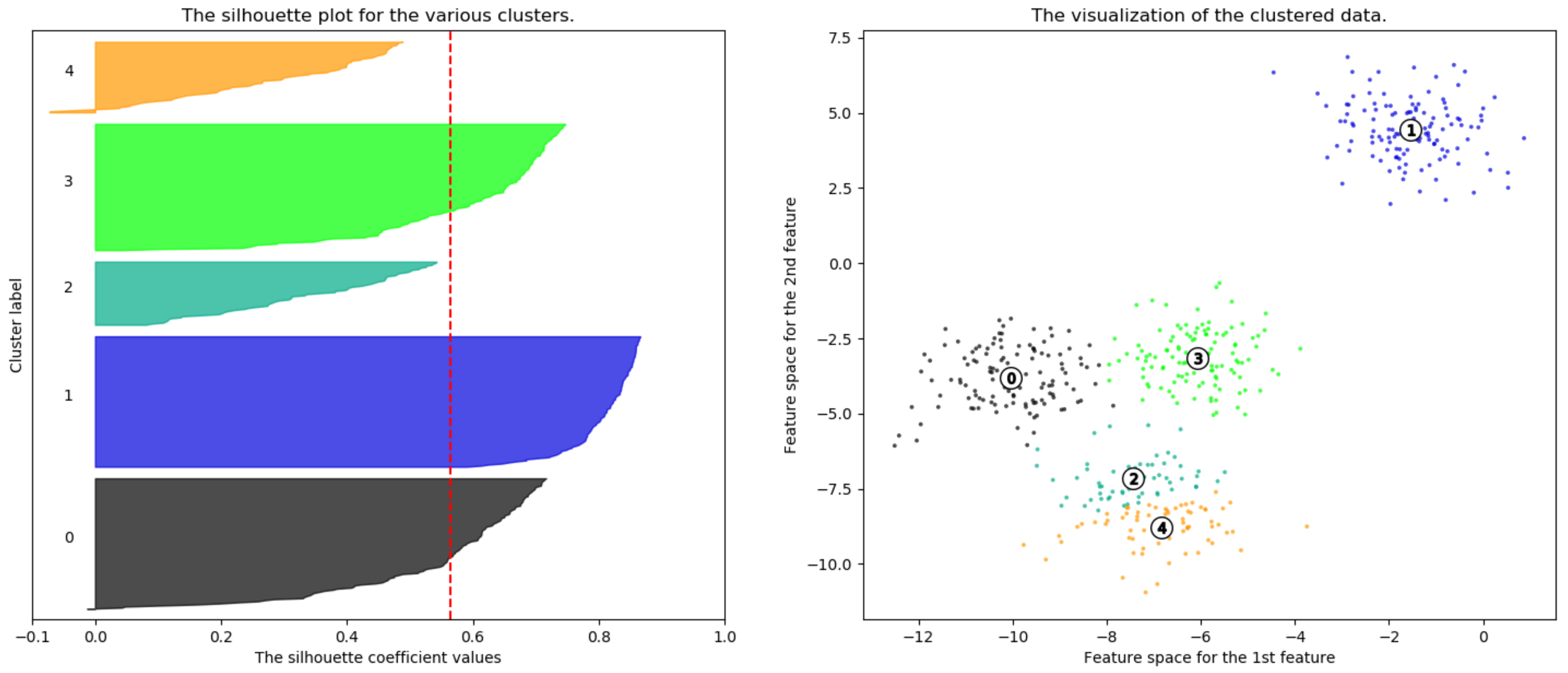


Silhouette analysis for KMeans clustering on sample data with n_clusters = 4

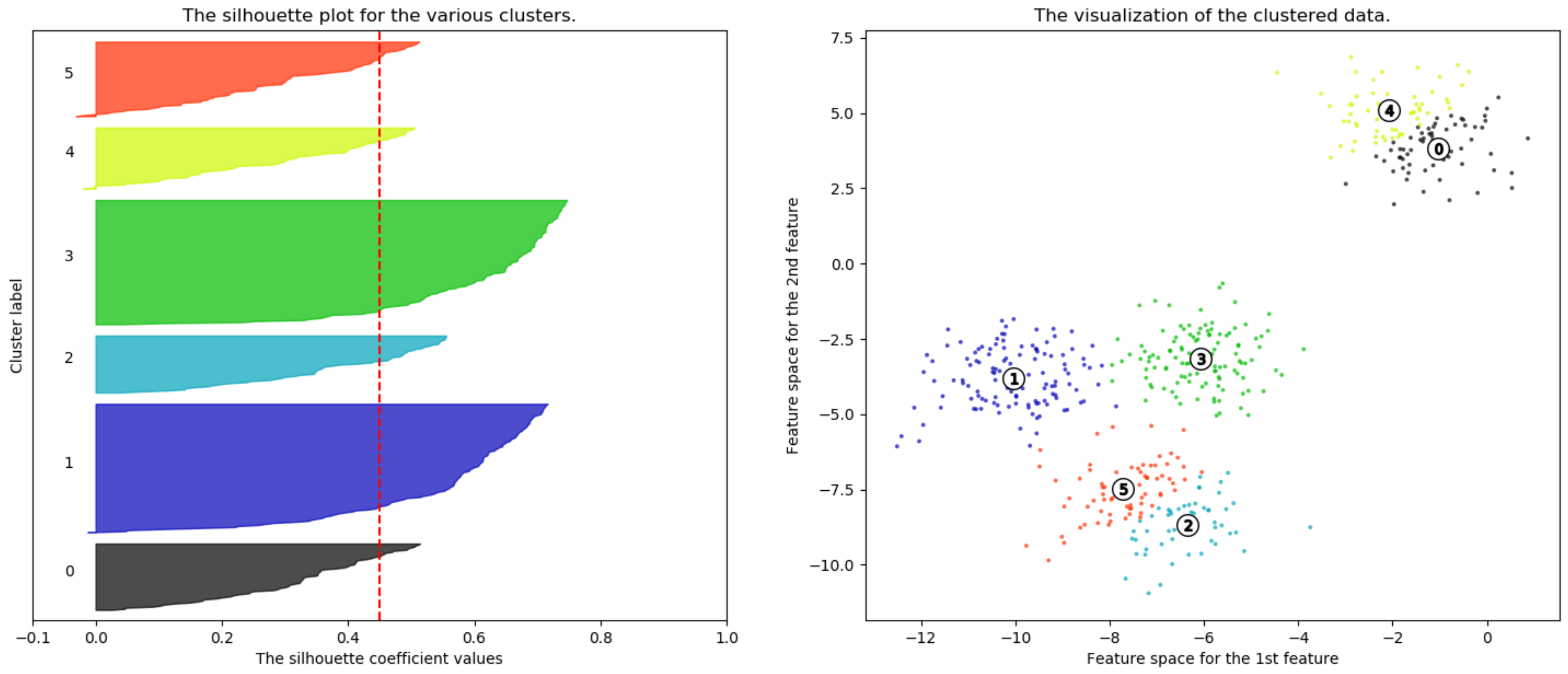


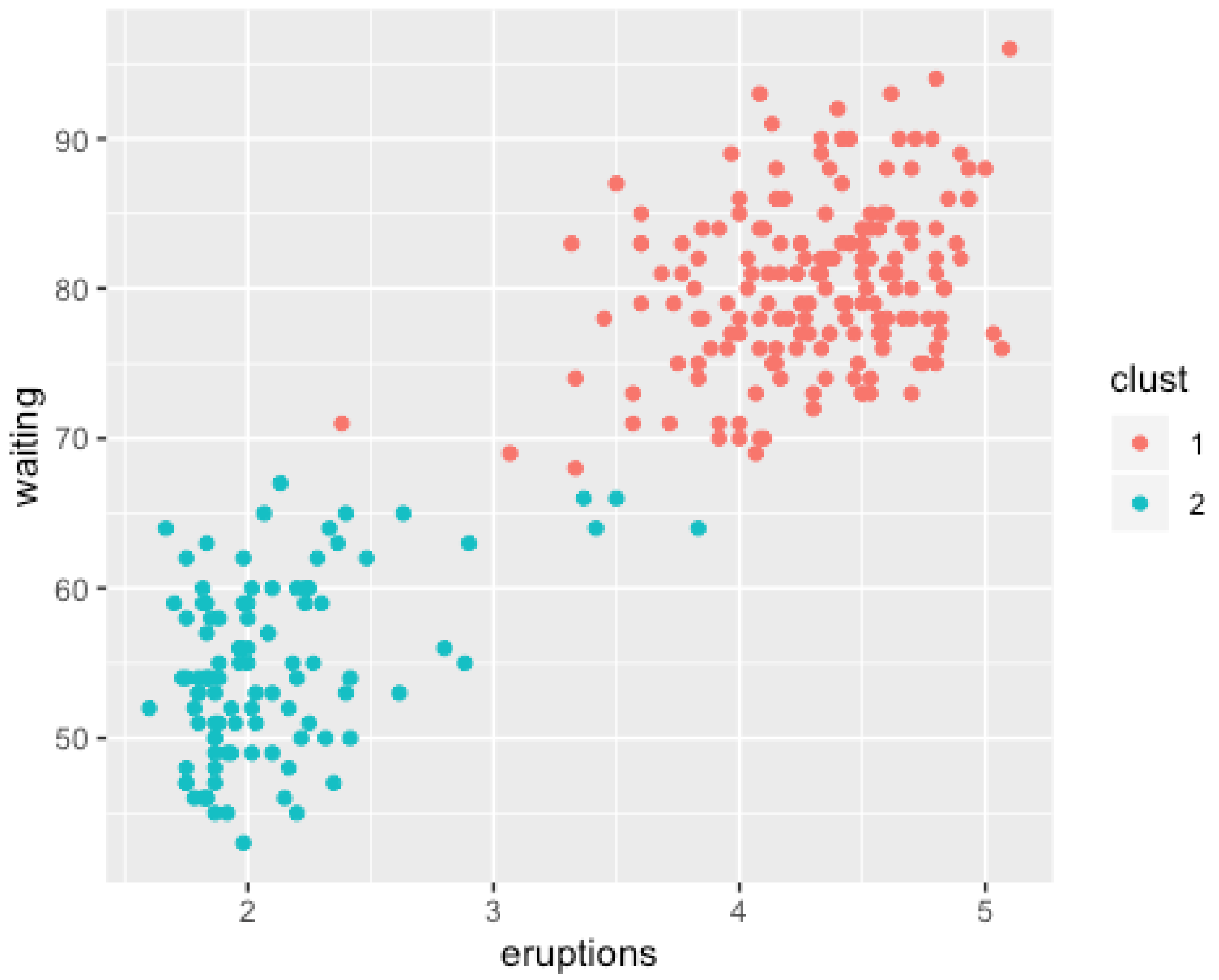


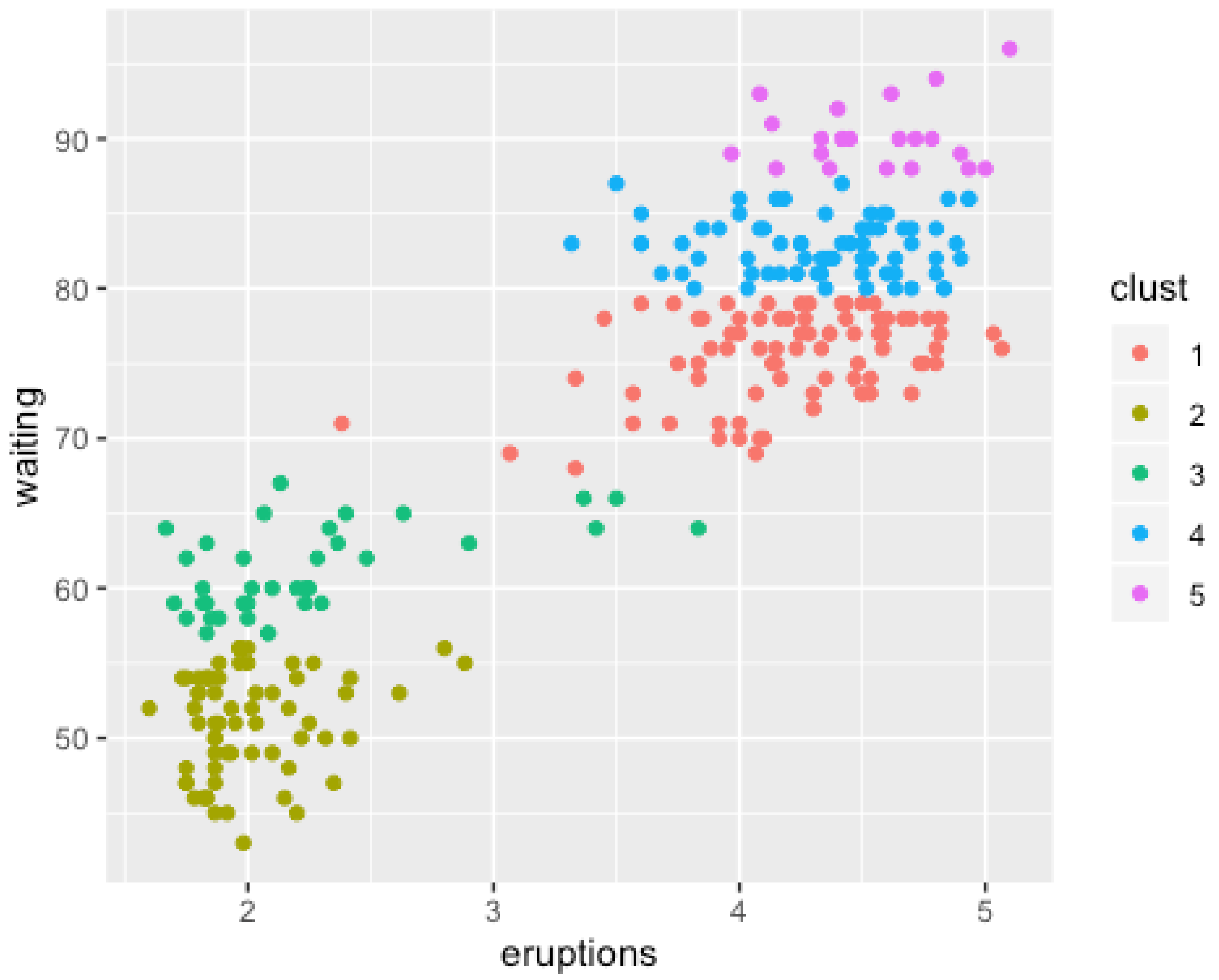
Silhouette analysis for KMeans clustering on sample data with n_clusters = 5

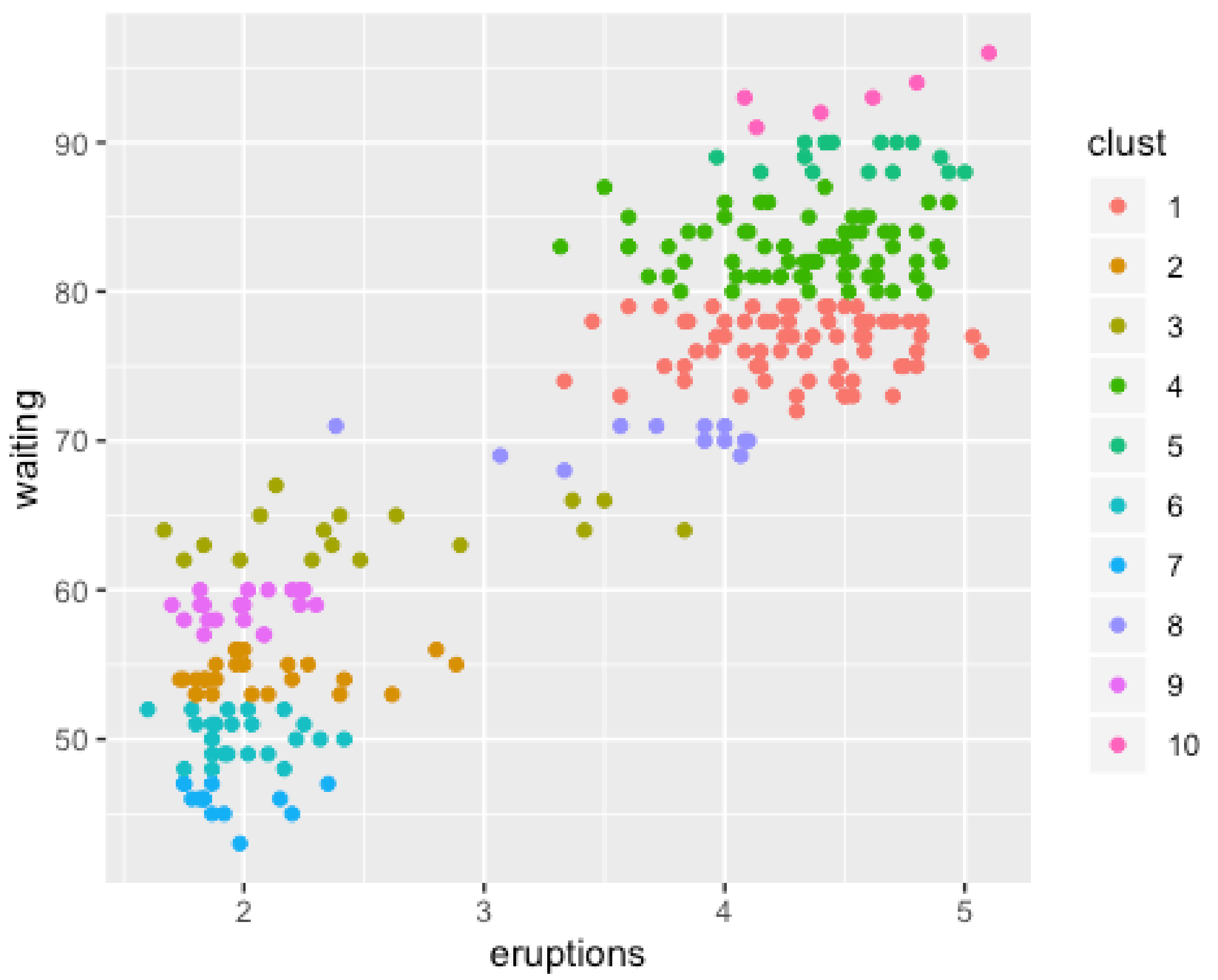


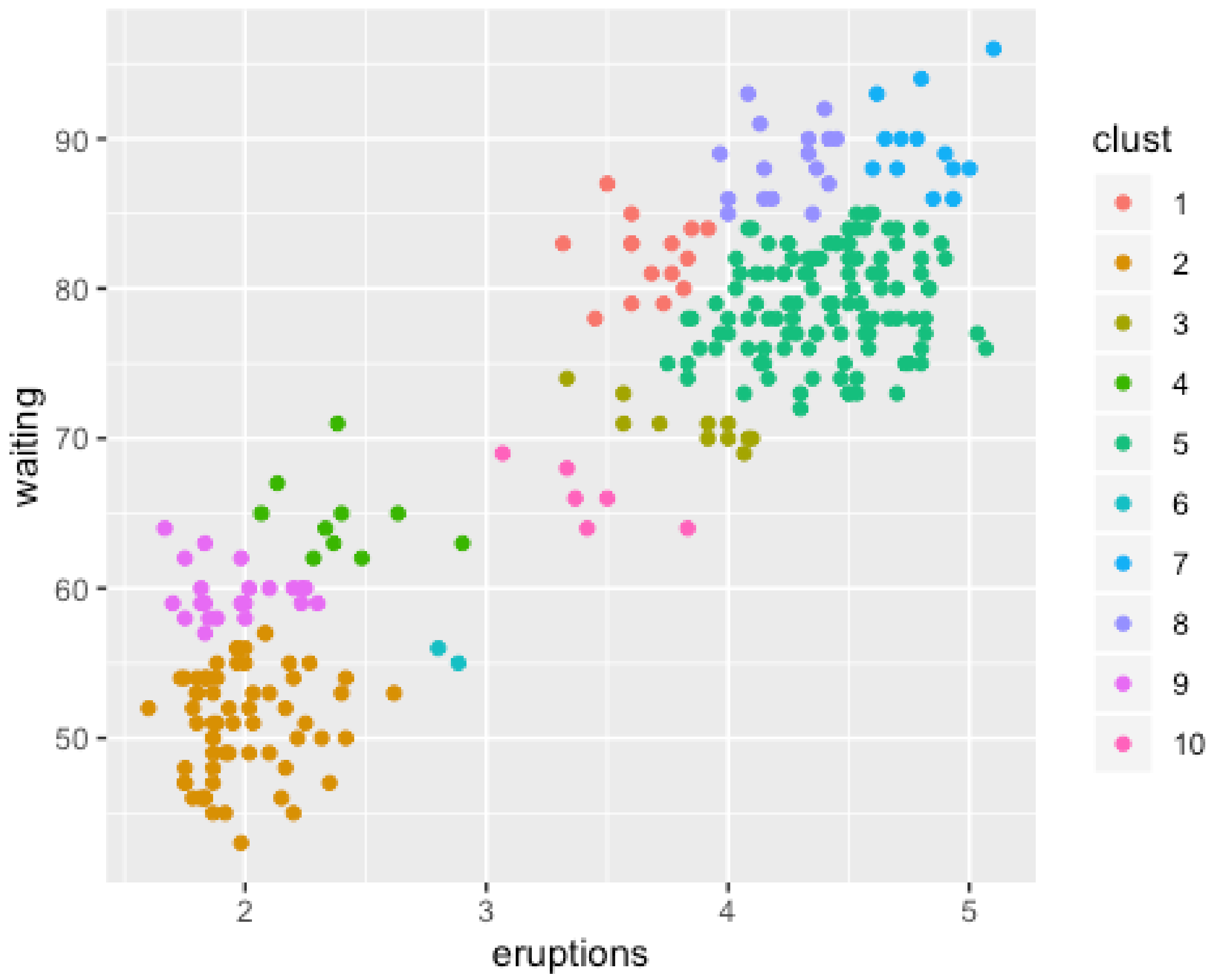
Silhouette analysis for KMeans clustering on sample data with n_clusters = 6

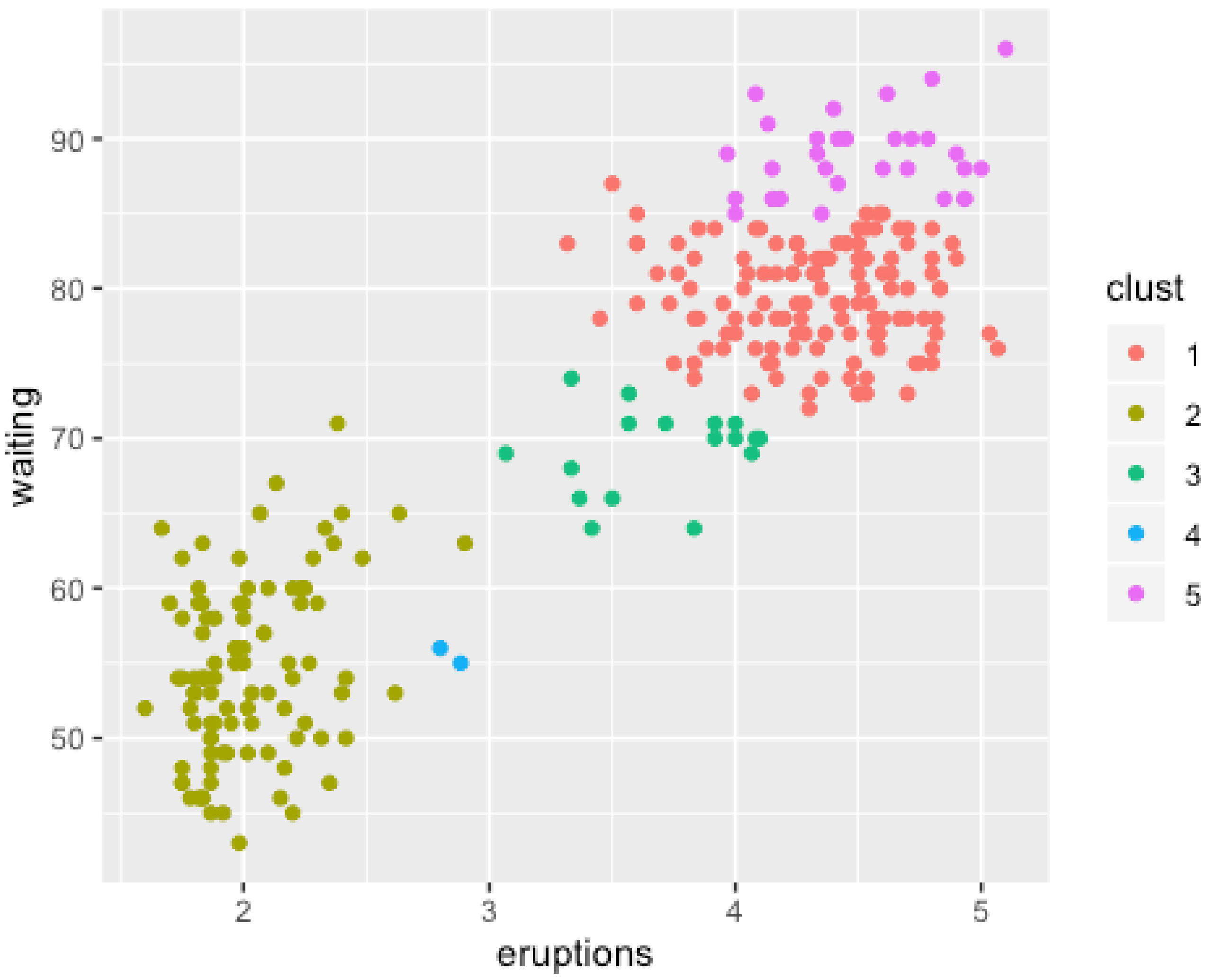


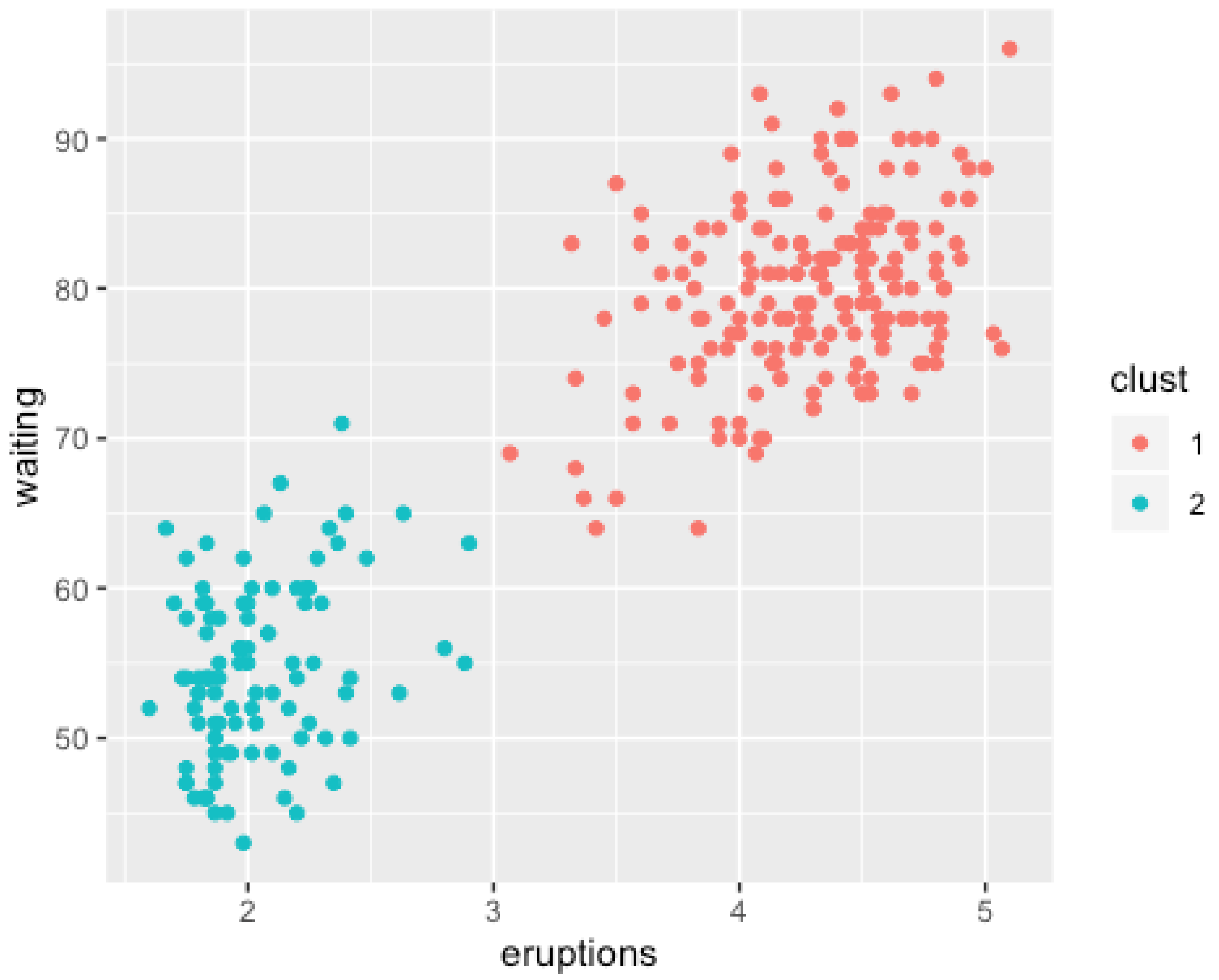

waiting
90
80
70
60
50
2
3
4
5
eruptions
clust
1
2


waiting
90
80
70
60
50
eruptions
2
3
4
5
clust
1
2
3
4
5


waiting
eruptions
90
80
70
60
50
2
3
4
5
clust
1
2
3
4
5
6
7
8
9
10


waiting
eruptions
90
80
70
60
50
2
3
4
5
clust
1
2
3
4
5
6
7
8
9
10


waiting
90
80
70
60
50
2
3
4
5
eruptions
clust
1
2
3
4
5


waiting
90
80
70
60
50
2
3
4
5
eruptions
clust
1
2

Summary of *k*-means clustering

Fast way of partitioning data into K clusters

It minimizes the sum of squared Euclidean distances to the clusters centroids

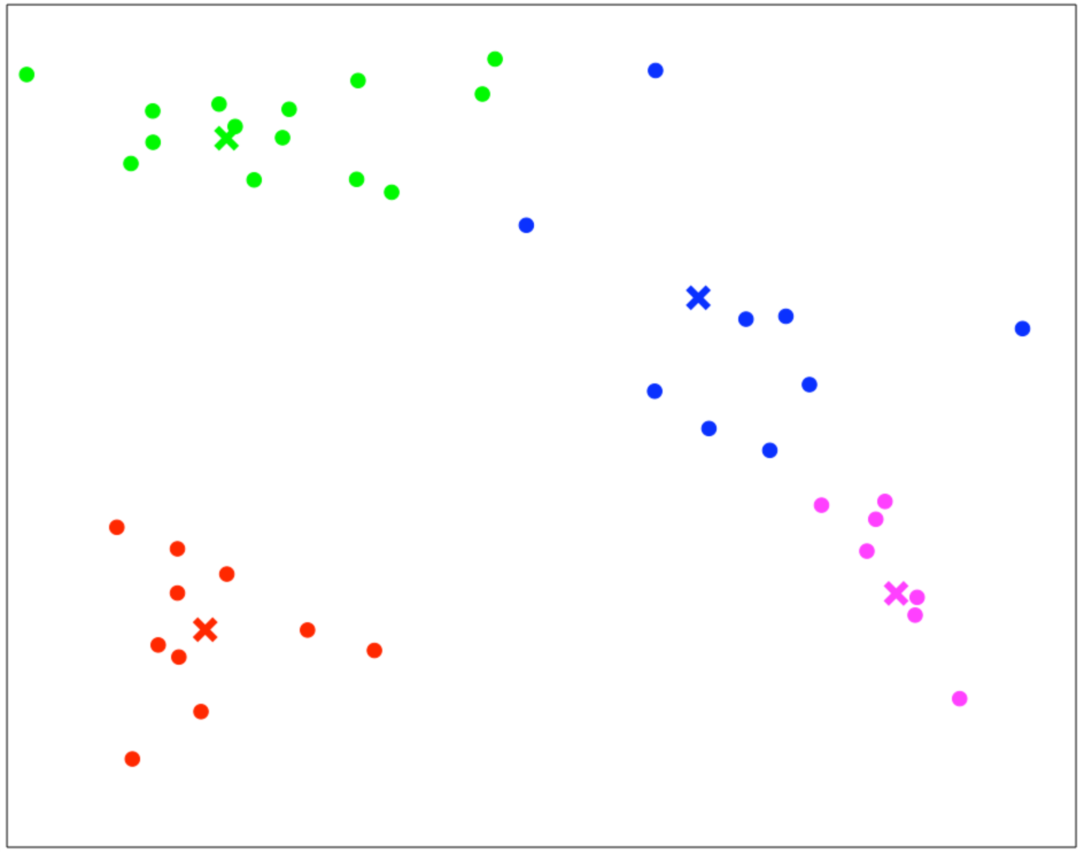
Different clusterings can result from different initializations

Natural way to add new points to existing clusters

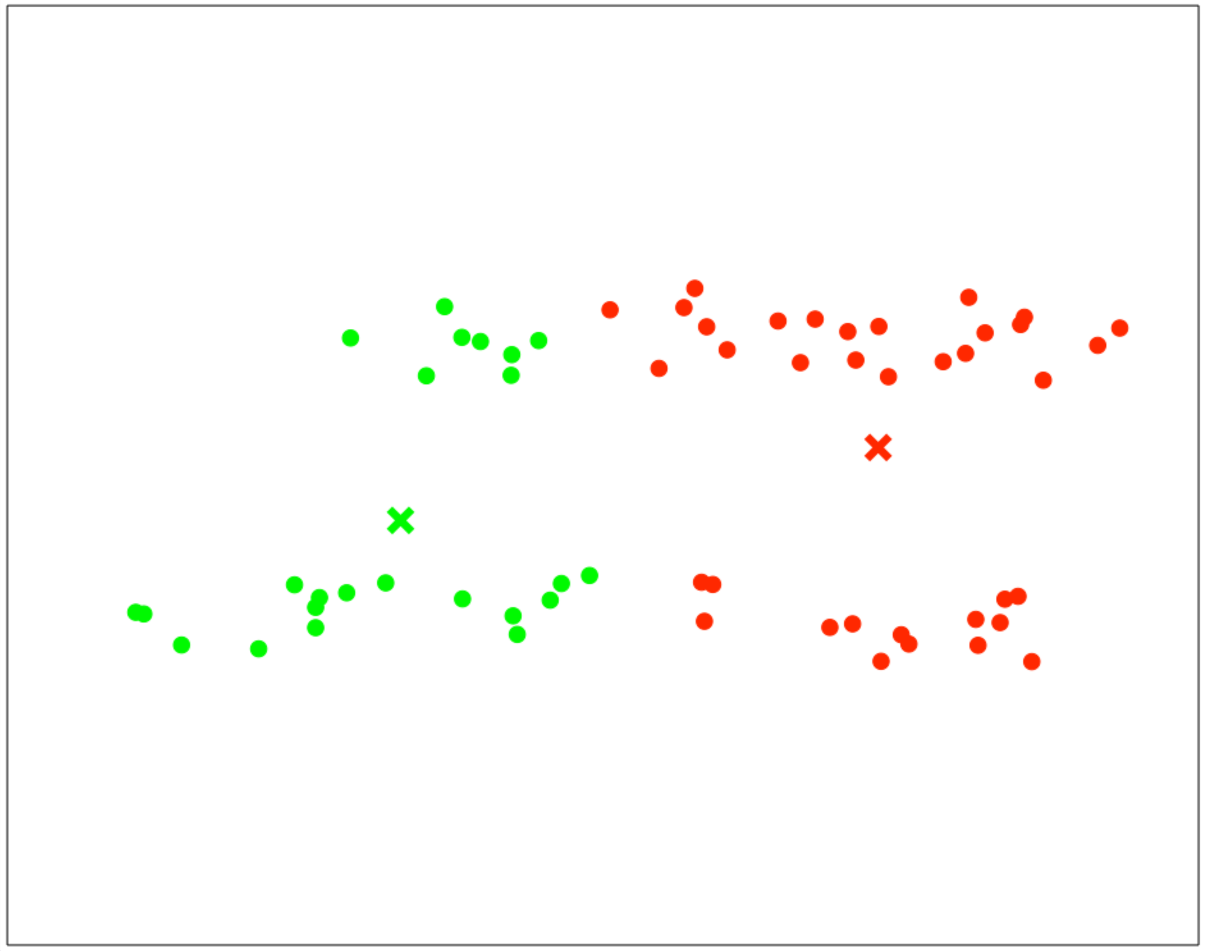
Bias toward round-ish, equally-sized clusters

Agglomerative Clustering

K-means: Nice, round clusters



K-means: Nice, round clusters



Agglomerative clustering

Input: Pairwise distances $d(\mathbf{x}, \mathbf{x}')$ between a set of data objects $\{\mathbf{x}_i\}$.

Output: A hierarchical clustering

- Assign each instance as its own cluster on a working list W.
- Repeat
  - Find the two clusters in W that are most "similar".
  - Remove them from W.
  - Add their union to W.
- until W contains a single cluster with all the data objects.
- Return *all clusters* appearing in W at any stage of the algorithm.

How many clusters after iteration *k*?

Answer: $n - k$, where n is the number of data objects.

Why?

- The working list W starts with n singleton clusters
- Each iteration removes two clusters from W and adds one new cluster
- The algorithm stops when W has one cluster, which is after $k = n - 1$ iterations

How do we measure dissimilarity between clusters?

Distance between nearest objects (“Single-linkage” agglomerative clustering, or “nearest neighbor”):

- $D(C, C') = \min_{\mathbf{x} \in C, \mathbf{x}' \in C'} d(\mathbf{x}, \mathbf{x}')$

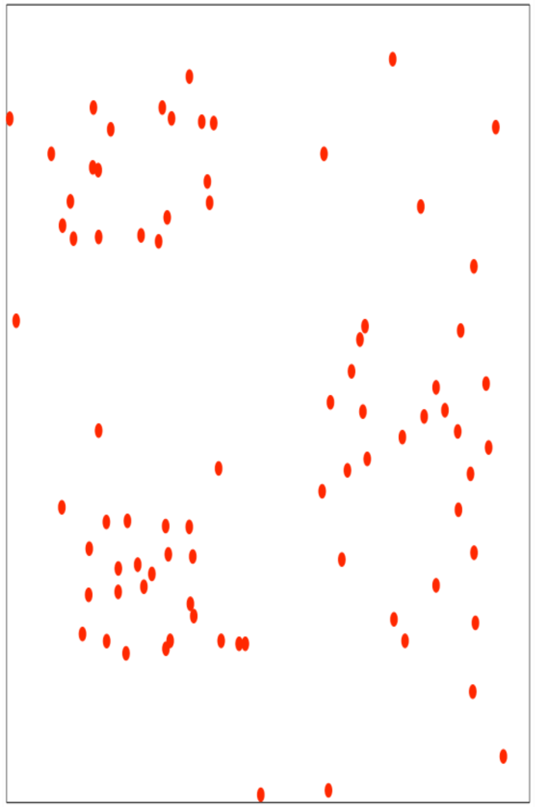
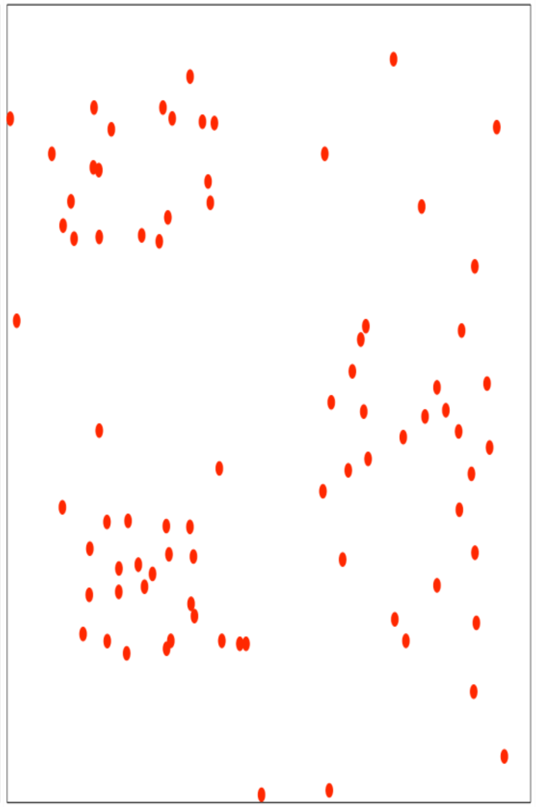
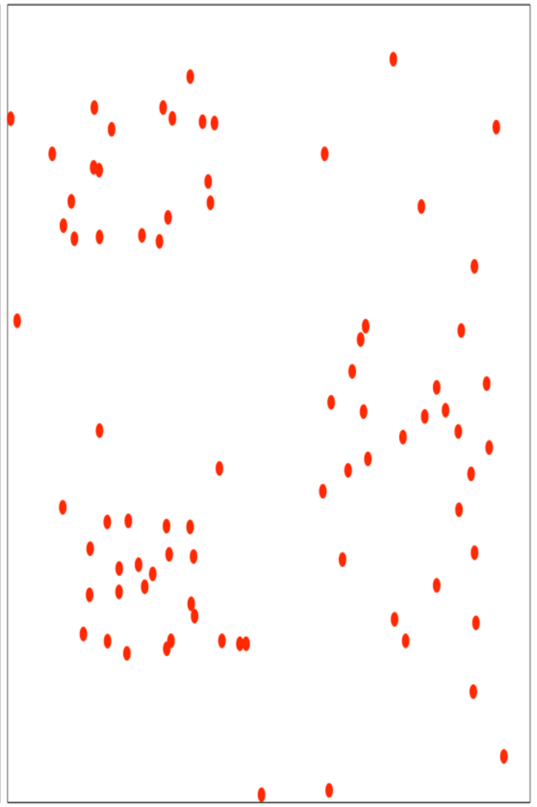
Average distance between objects (“Group-average” agglomerative clustering):

- $D(C, C') = \frac{1}{|C||C'|} \sum_{\mathbf{x} \in C, \mathbf{x}' \in C'} d(\mathbf{x}, \mathbf{x}')$

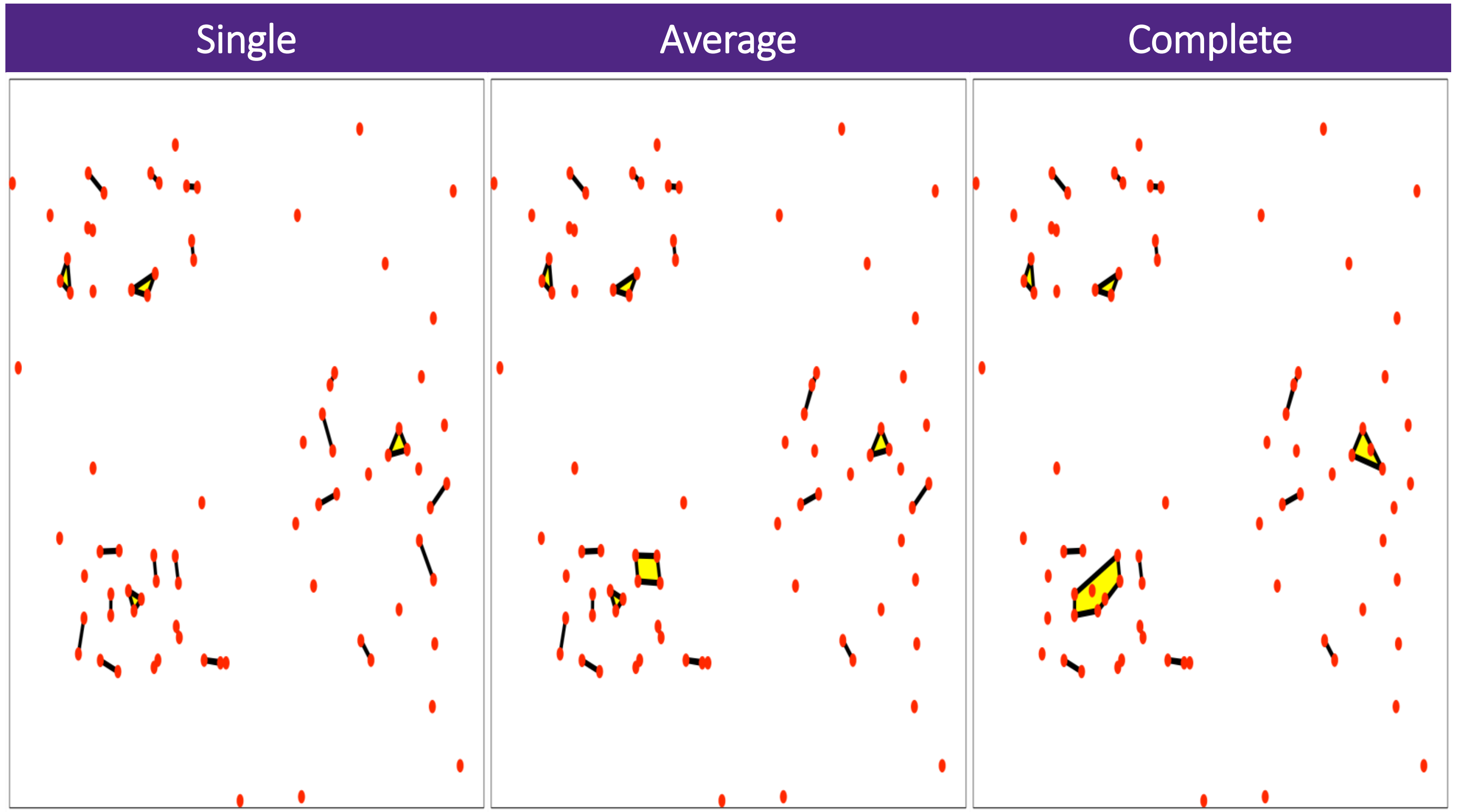
Distance between farthest objects (“Complete-linkage” agglomerative clustering, or “furthest neighbor”):

- $D(C, C') = \max_{\mathbf{x} \in C, \mathbf{x}' \in C'} d(\mathbf{x}, \mathbf{x}')$

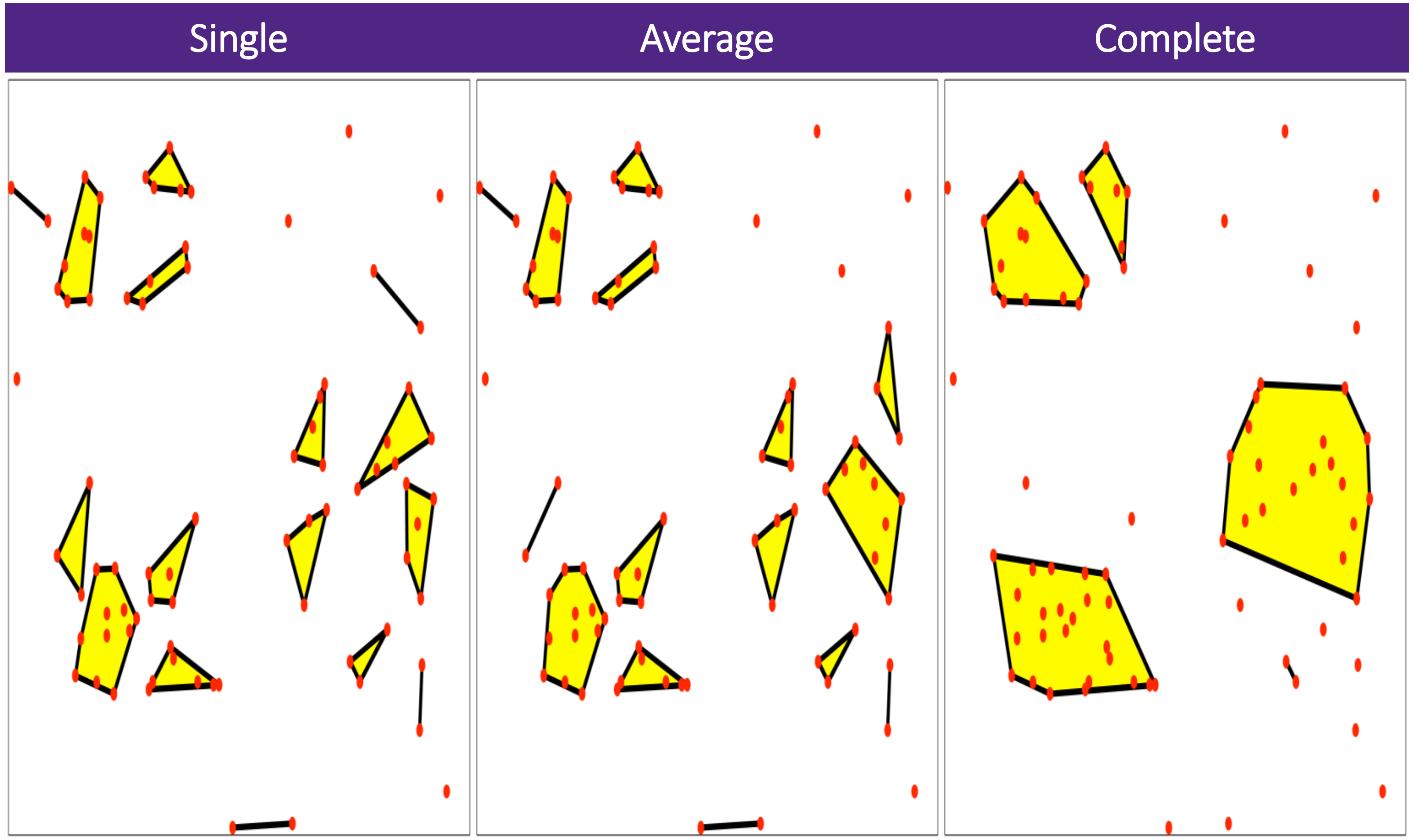
Example 1: Data

Single	Average	Complete
		

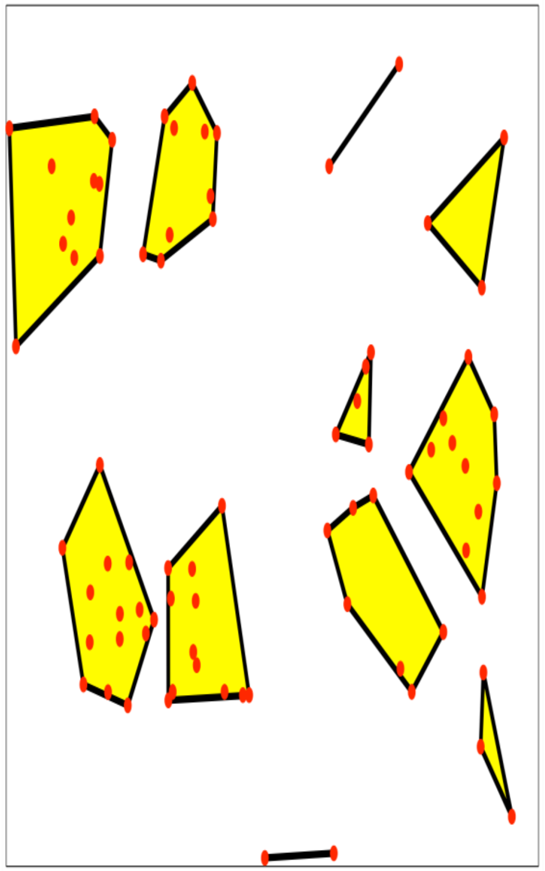
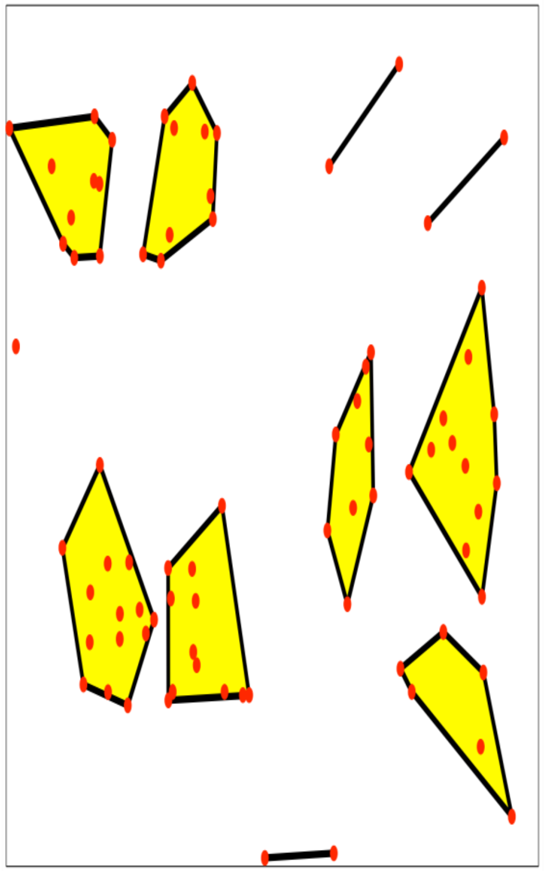
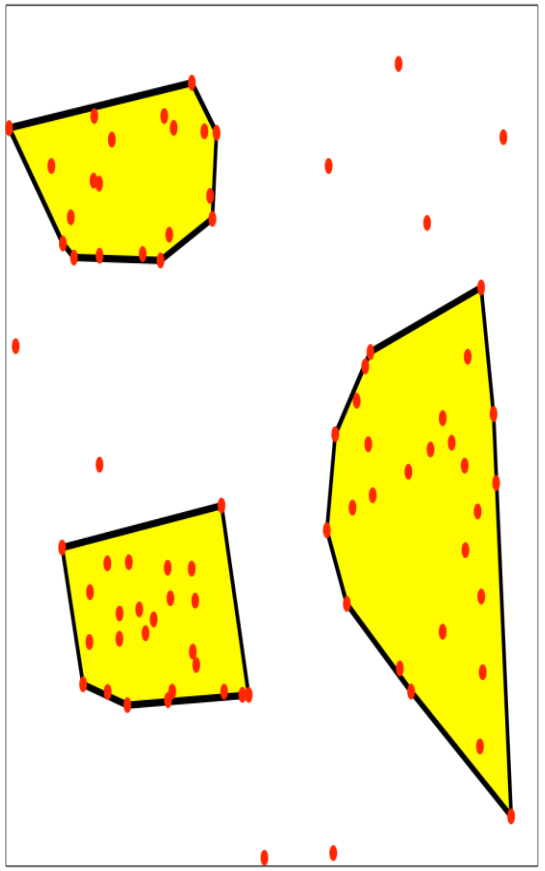
Example 1: Iteration 30



Example 1: Iteration 60

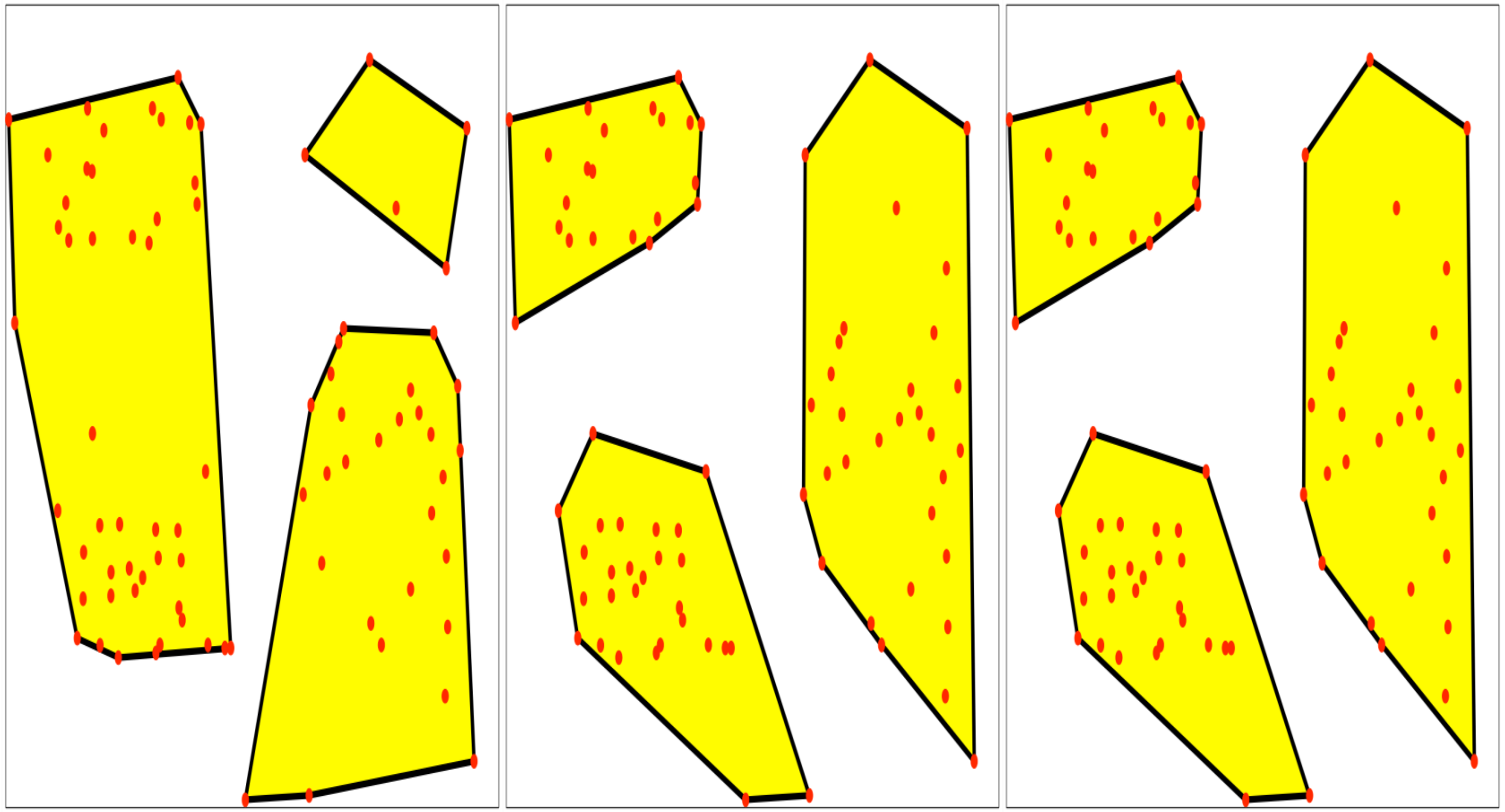


Example 1: Iteration 70

Single	Average	Complete
		

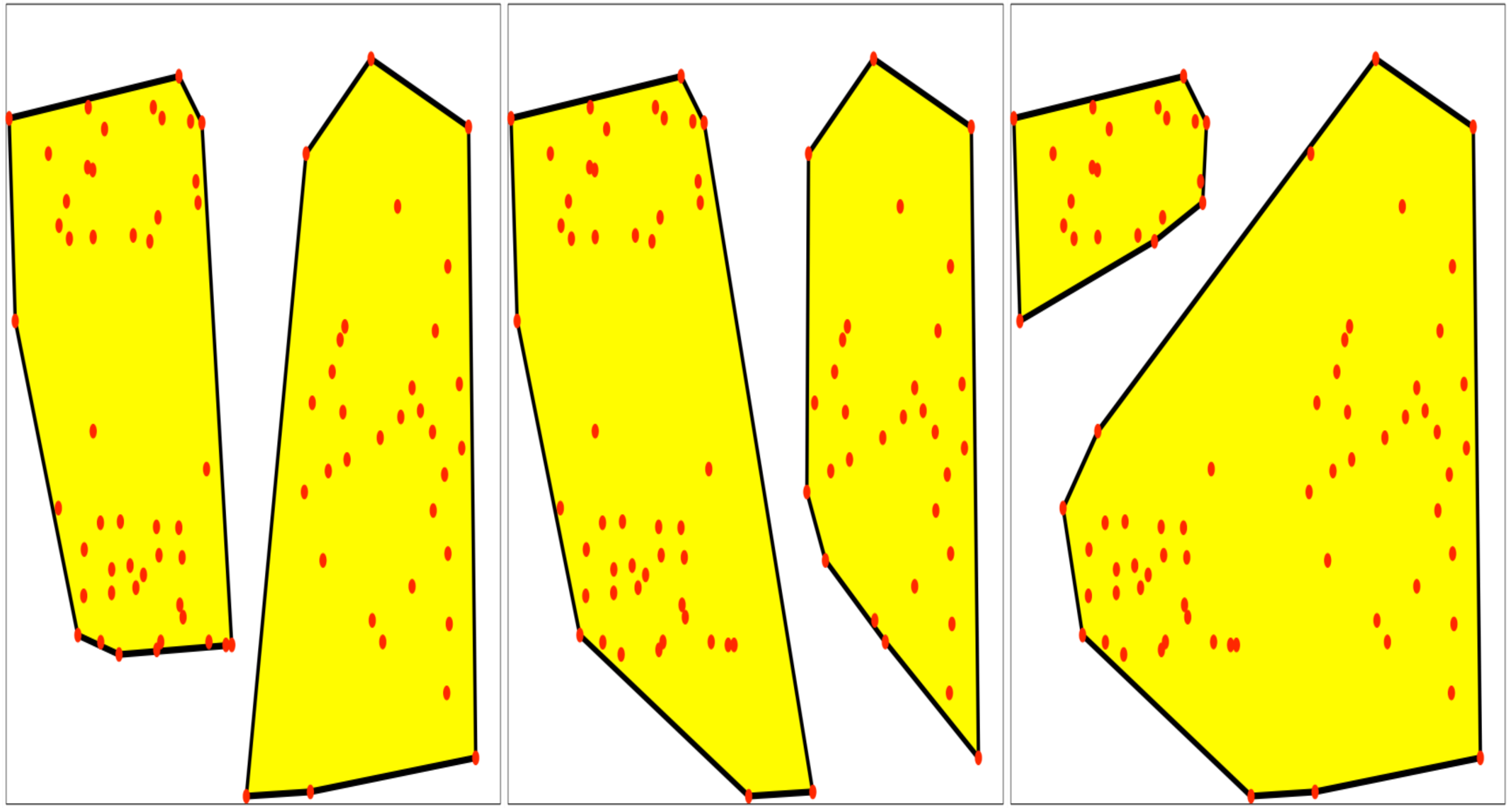
Example 1: Iteration 78

Single	Average	Complete

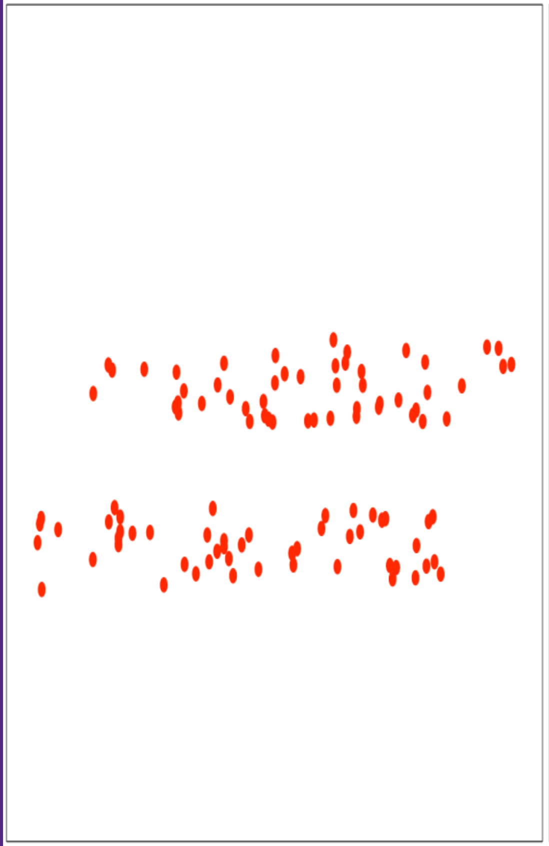
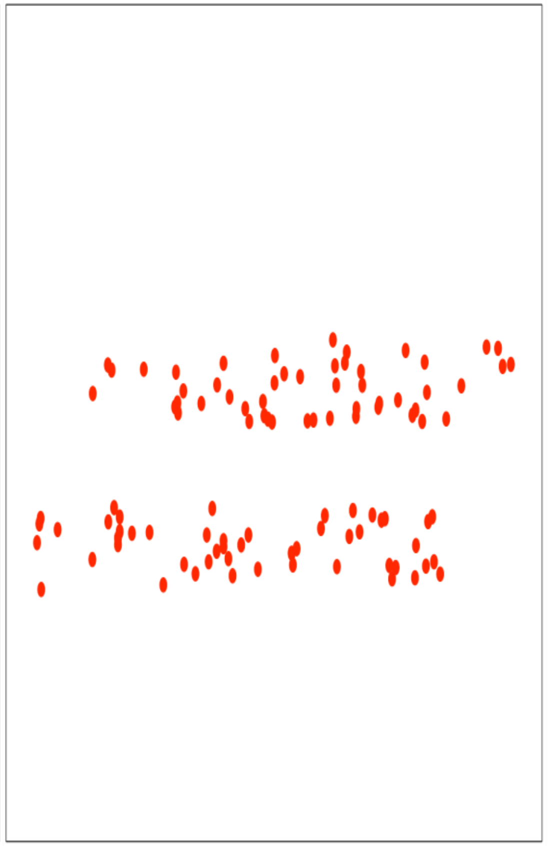
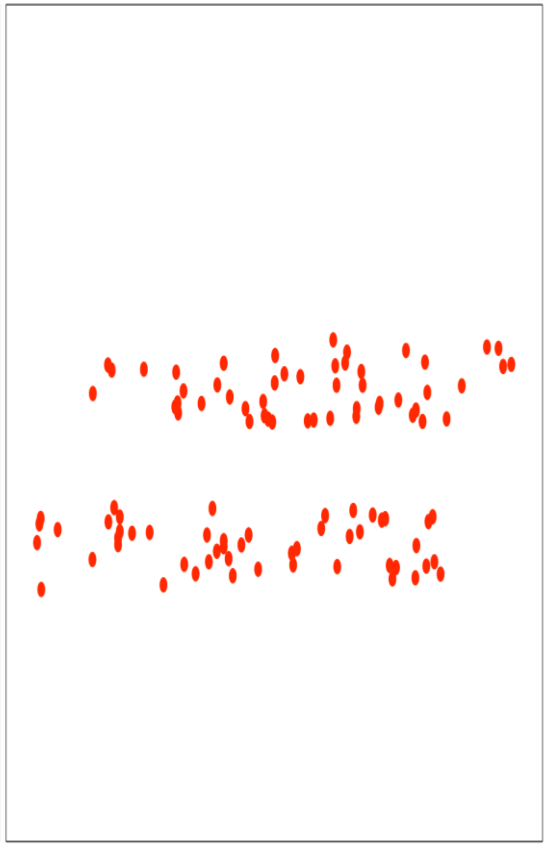


Example 1: Iteration 79

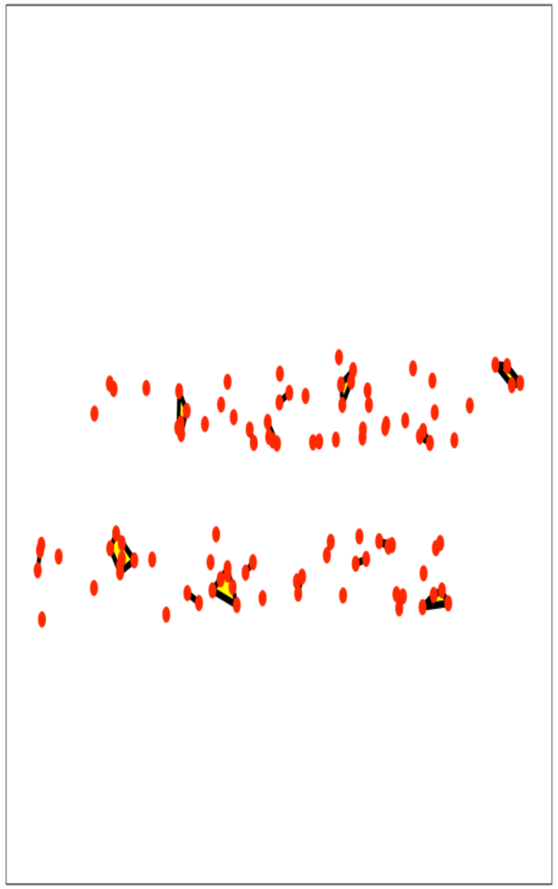
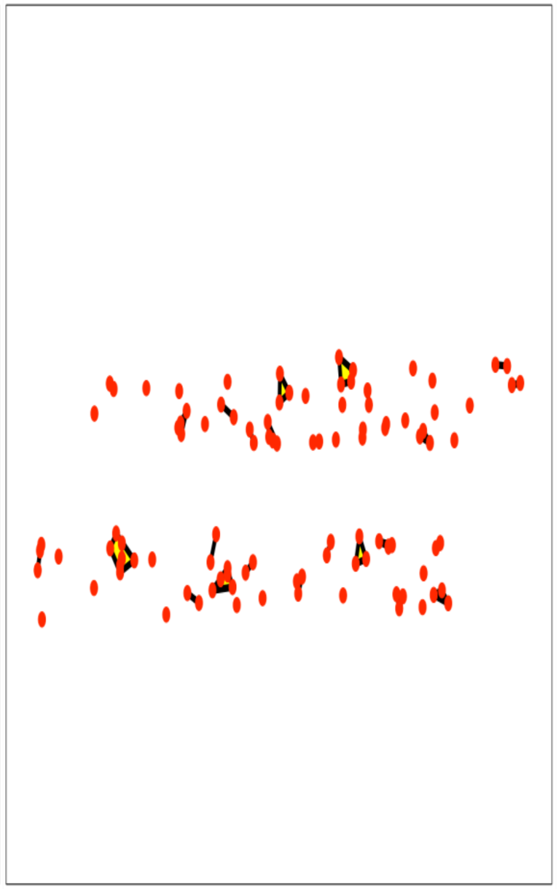
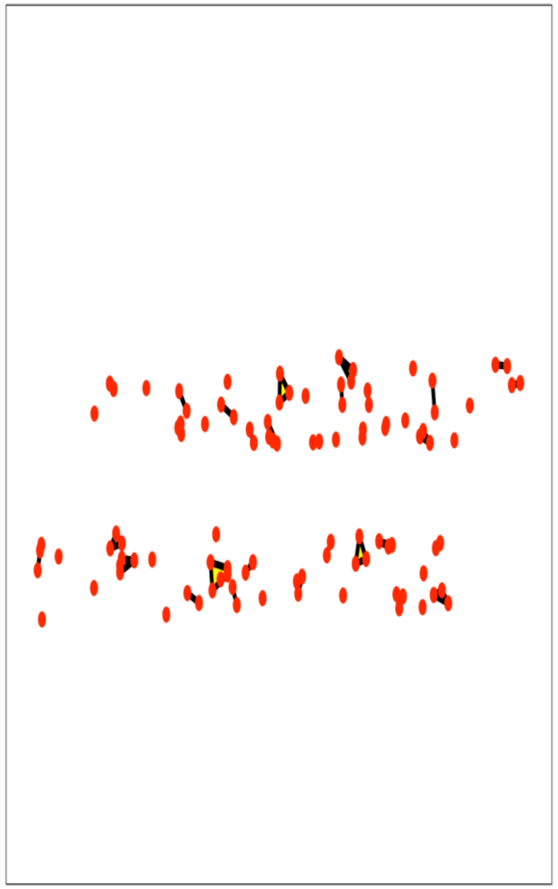
Single	Average	Complete



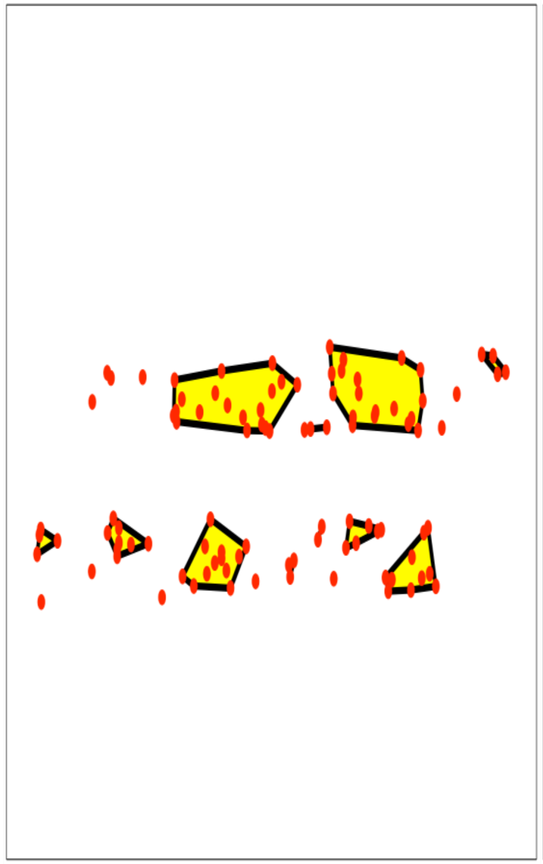
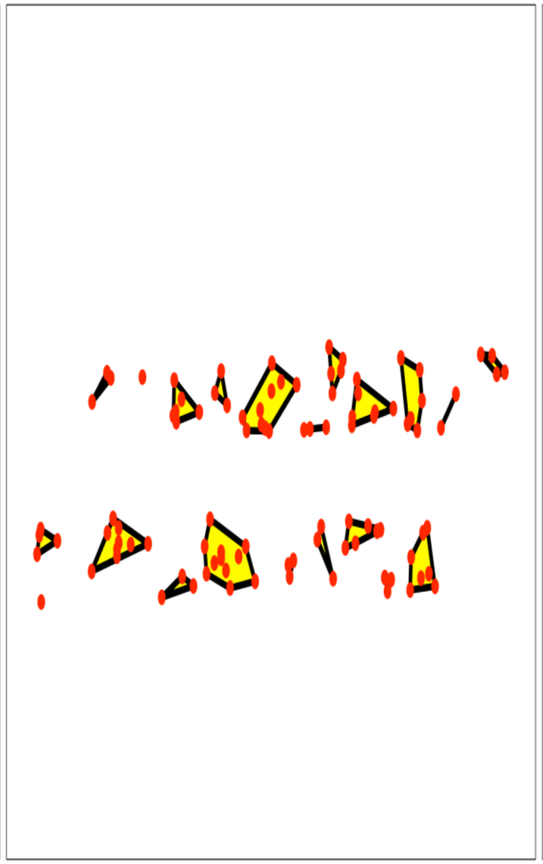
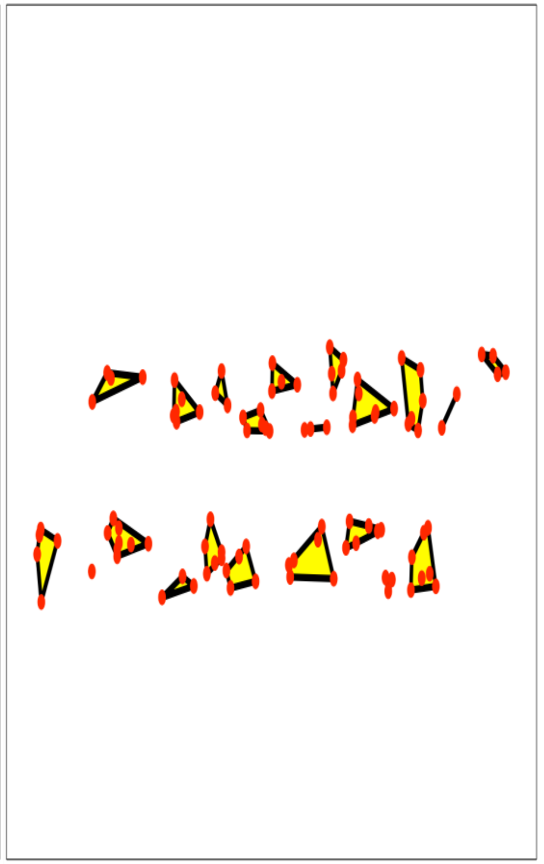
Example 2: Data

Single	Average	Complete
		

Example 2: Iteration 50

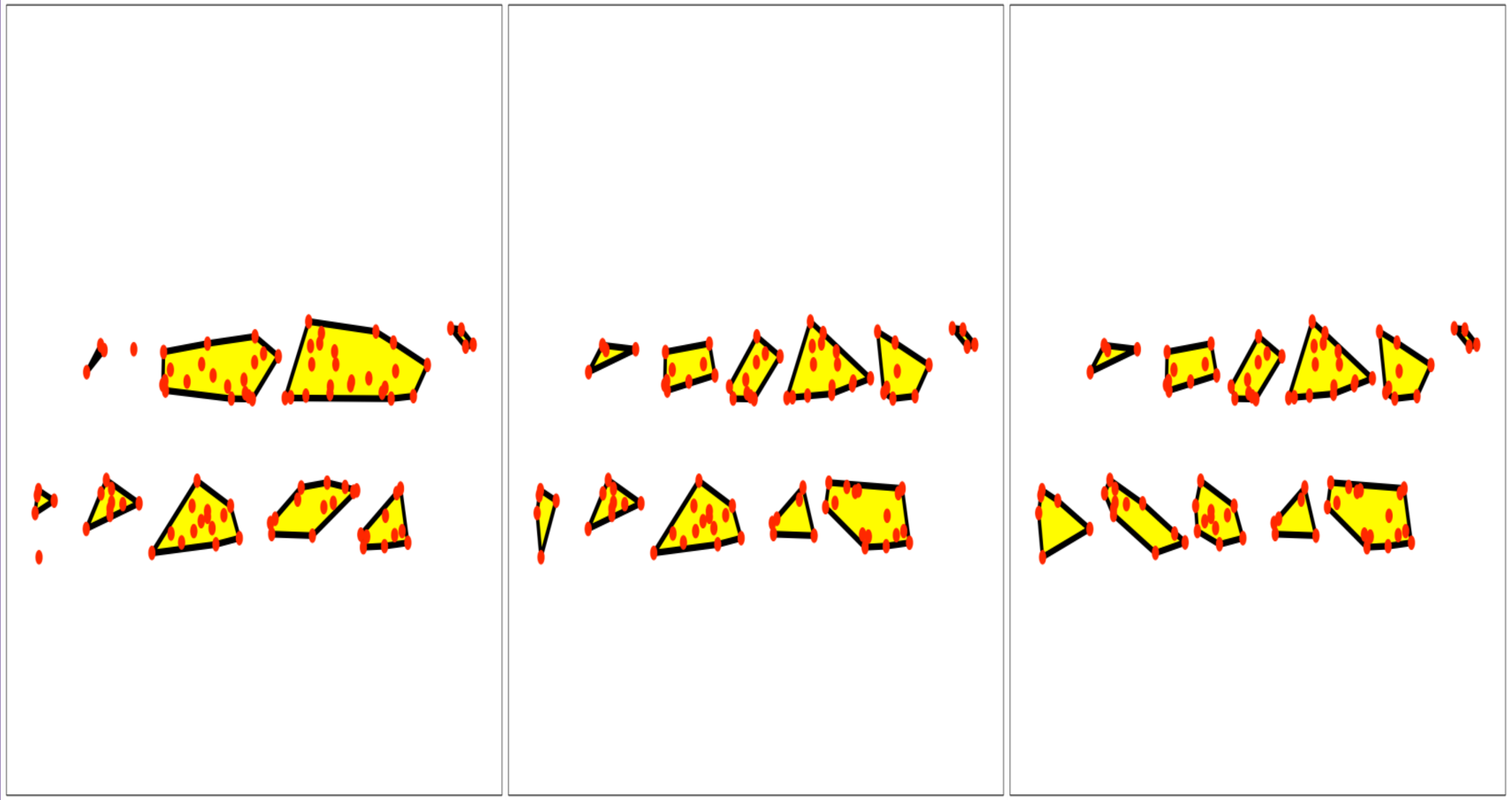
Single	Average	Complete
		

Example 2: Iteration 80

Single	Average	Complete
		

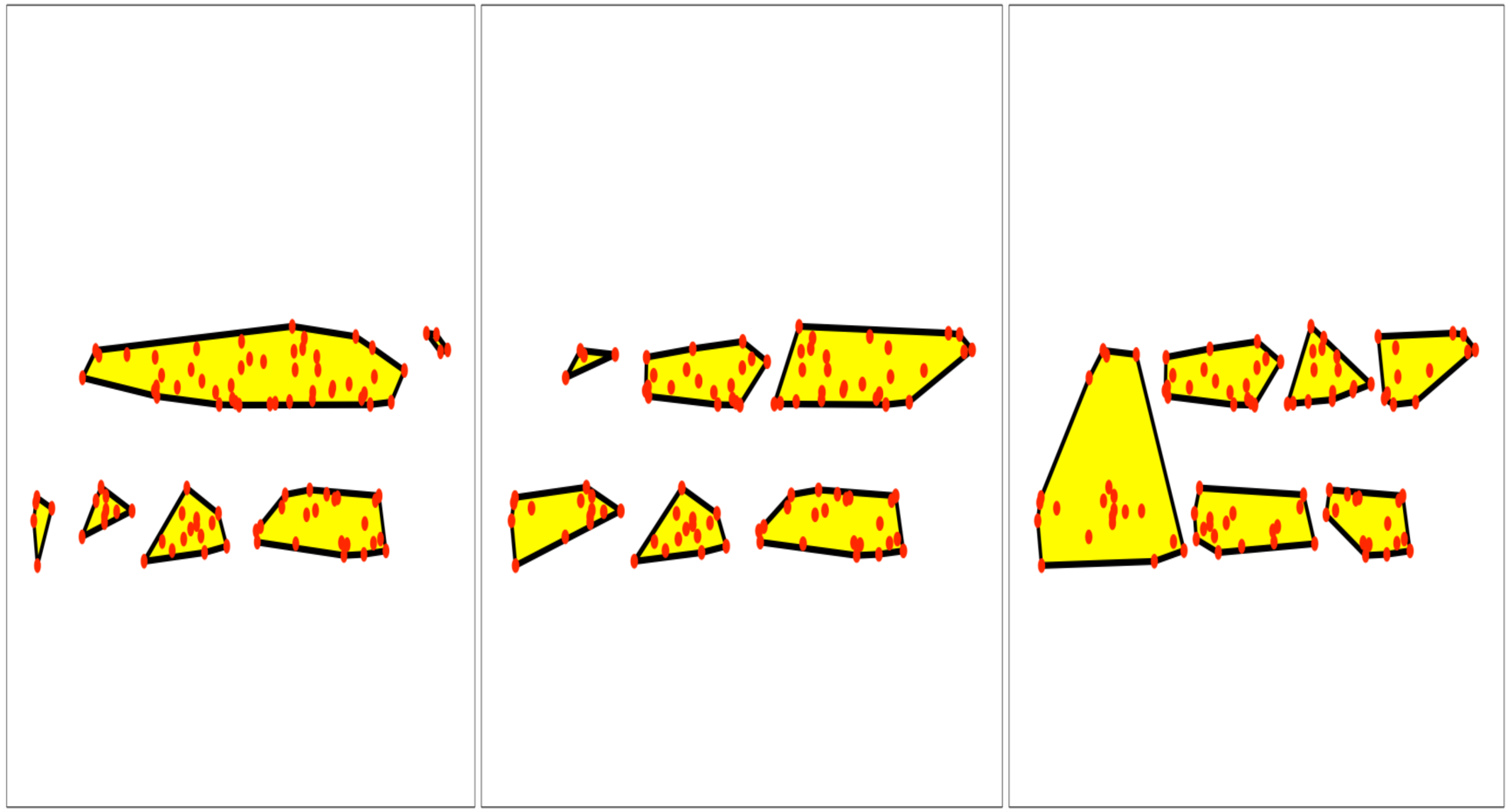
Example 2: Iteration 90

Single	Average	Complete

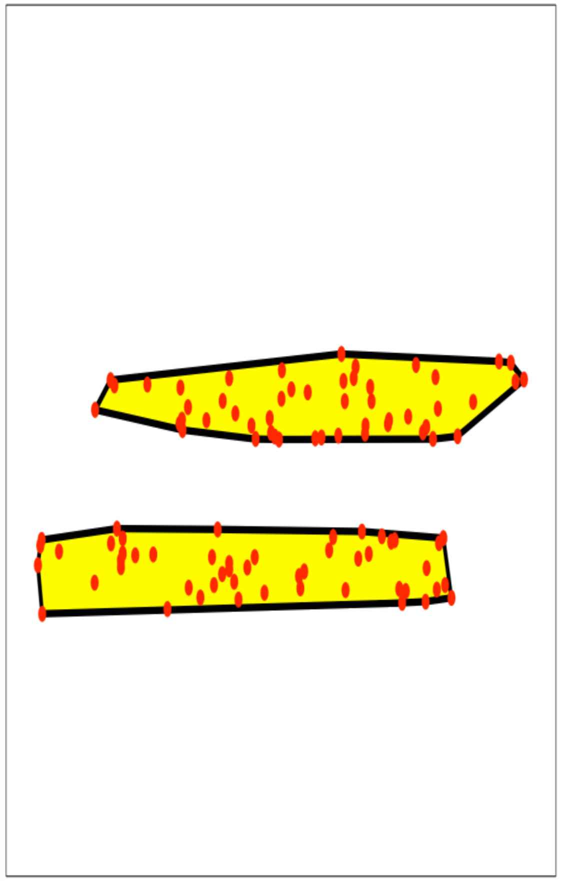
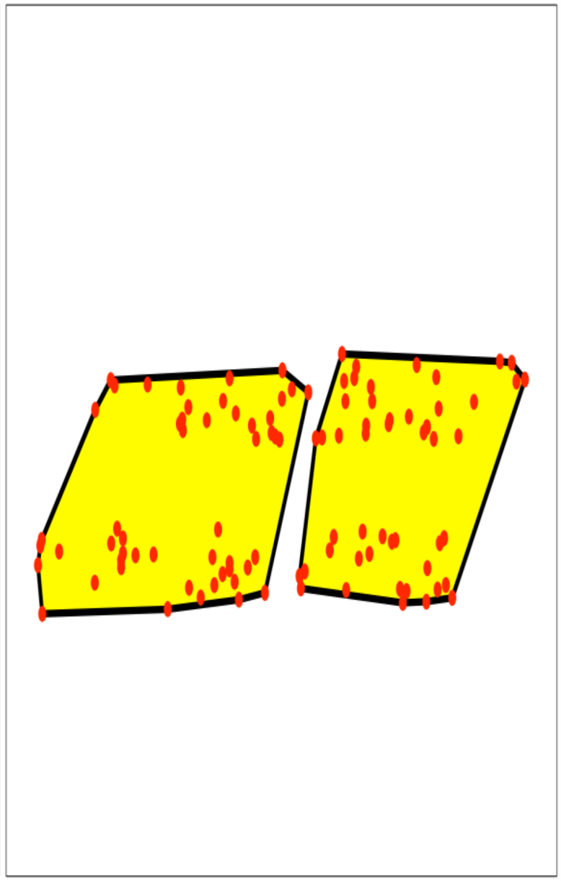
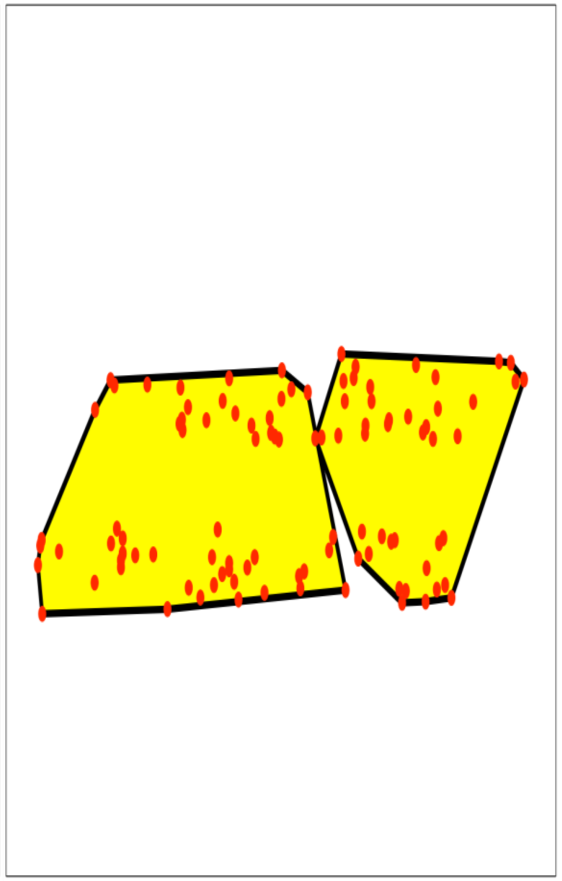


Example 2: Iteration 95

Single	Average	Complete



Example 2: Iteration 99

Single	Average	Complete
		

Intuitions about cluster similarity

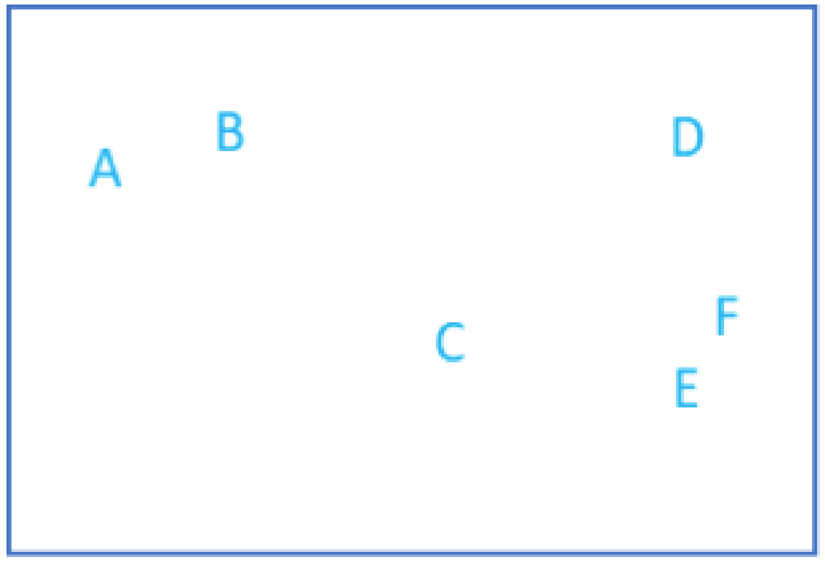
Single-linkage

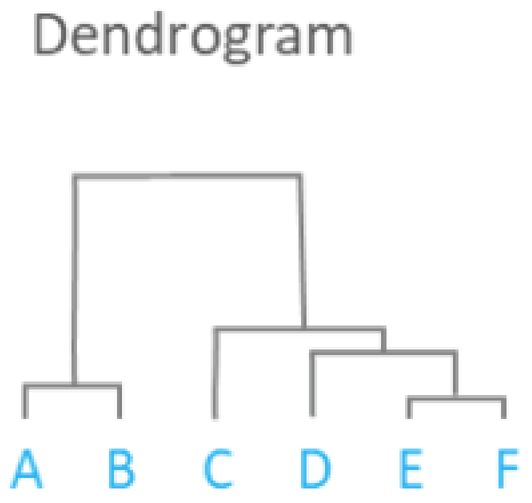
- Favors spatially-extended / filamentous clusters
- Often leaves singleton clusters until near the end

Complete-linkage favours compact clusters

Average-linkage is somewhere in between

Dendrograms





https://www.displayr.com/what-is-dendrogram/

Summary of Hierarchical clustering

Organizes data objects into a tree based on similarity.

Agglomerative (bottom-up) tree construction is most popular.

There are several choices of distance metric (linkage criterion)

No natural way to find which cluster a new point should belong to

Practical considerations

It is a good idea to scale before cluster.

That way every we satisfy requirement that all variables weight the same.

Careful with artifacts introduced by this!

Try and decide, most of the time scaling will do the trick.

Summary

Distance-based learning

Supervised: k-nearest neighbours

Unsupervised: k-means

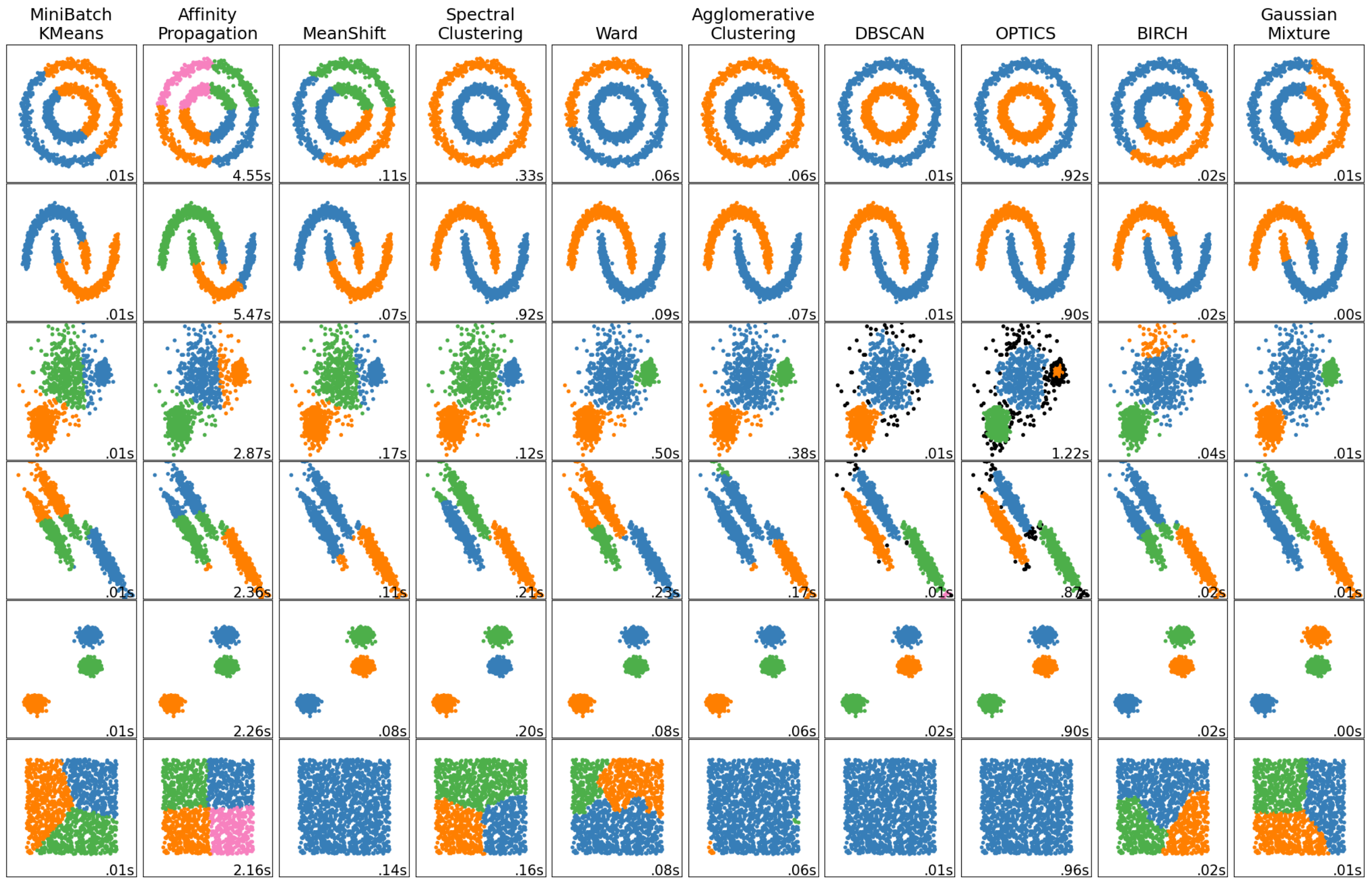
K-means

Non-deterministic, round, equally-sized clusters

Agglomerative

More control over cluster shape

Single, average, complete linkages give different results



https://scikit-learn.org/stable/modules/clustering.html