Unsupervised Learning Techniques

Chapter 9: pp 235 – 259

Motivation

- Very expensive to generate labeled data
- Majority of available data is unlabeled

Unsupervised Learning

- Clustering
  - Groups similar instances into clusters
- Anomaly detection
  - Learns what is “normal” to detect abnormal instances

Clustering

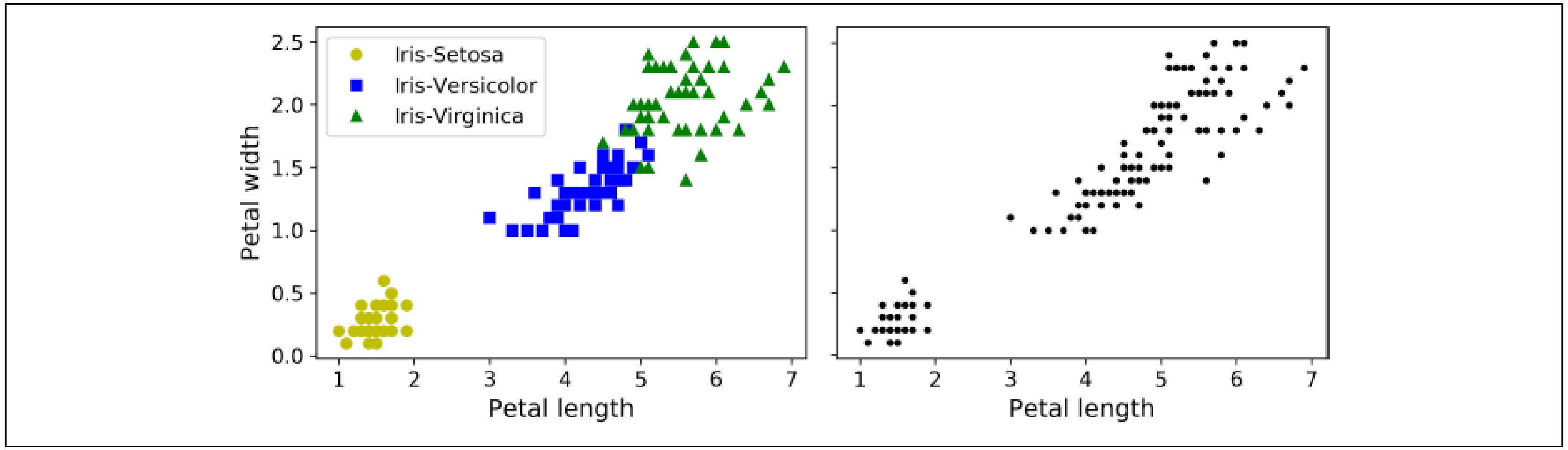


Figure 9-1. Classification (left) versus clustering (right)

- Customer segmentation
- Data analysis
- As a dimensionality reduction technique
- Anomaly detection
- Semi-supervised learning

K-Means

- Specify the number of clusters $k = 5$

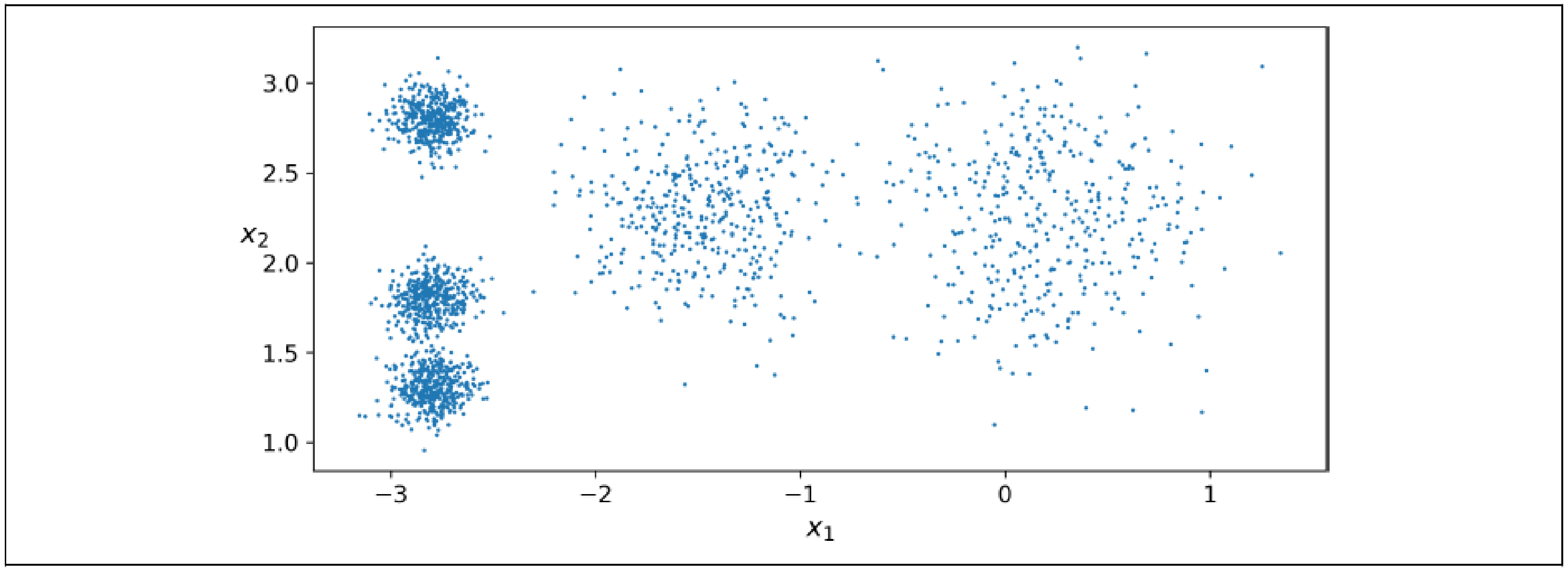


Figure 9-2. An unlabeled dataset composed of five blobs of instances

- Algorithm assigns cluster label/index
  - Not the same as class label
- Algorithm also finds the centroids

```
>>> kmeans.cluster_centers_
array([[-2.80389616,  1.80117999],
       [ 0.20876306,  2.25551336],
       [-2.79290307,  2.79641063],
       [-1.46679593,  2.28585348],
       [-2.80037642,  1.30082566]])
```

Voronoi Tessellation

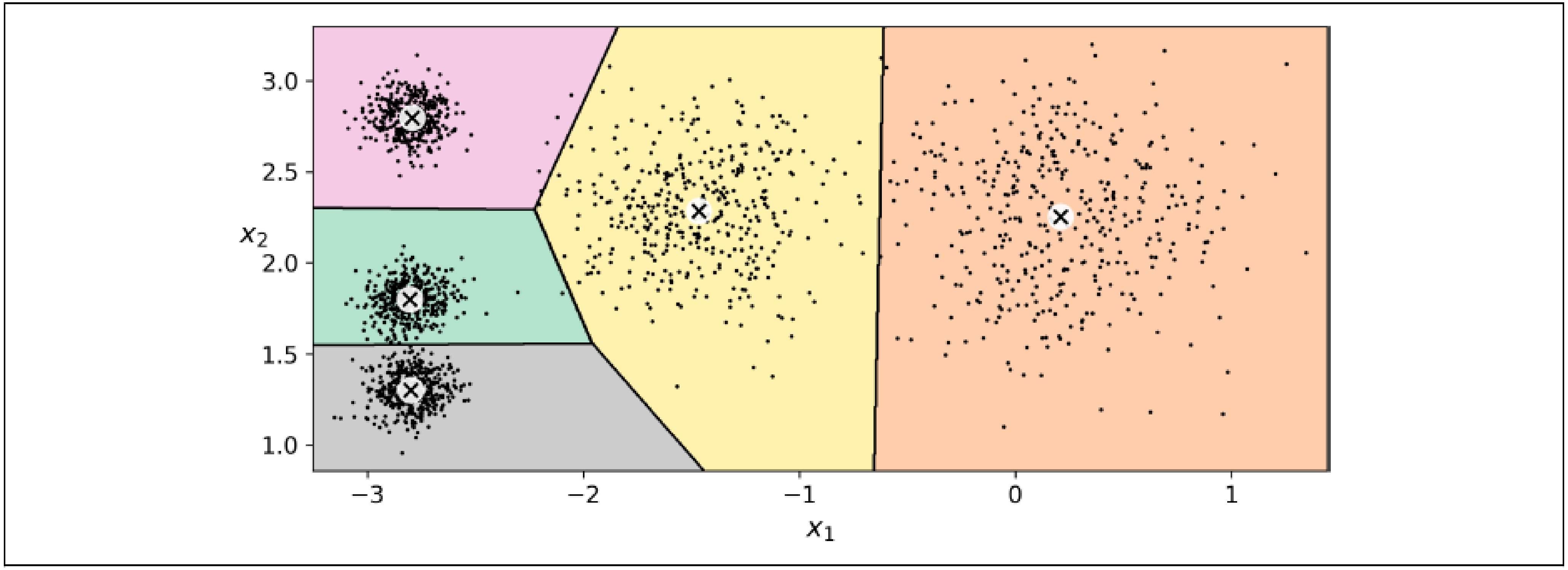


Figure 9-3. K-Means decision boundaries (Voronoi tessellation)

- Hard clustering vs soft clustering

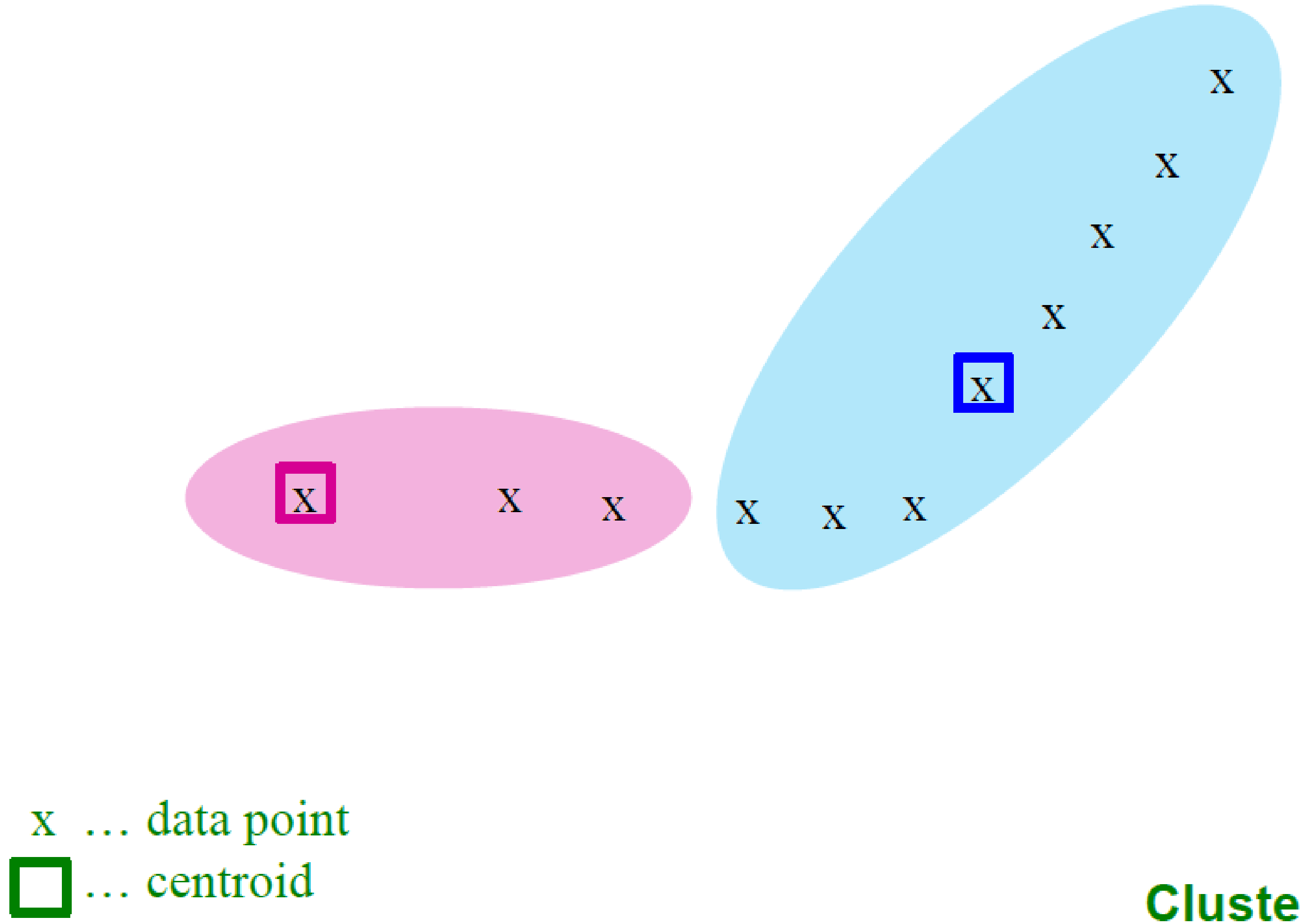
K-Means Algorithm

1. Randomly place the centroids
2. For each instance, find the nearest cluster and assign cluster index
3. For each cluster, recompute its centroid location by taking the mean of all the instances assigned to that cluster
4. Repeat steps 2 and 3 until convergence

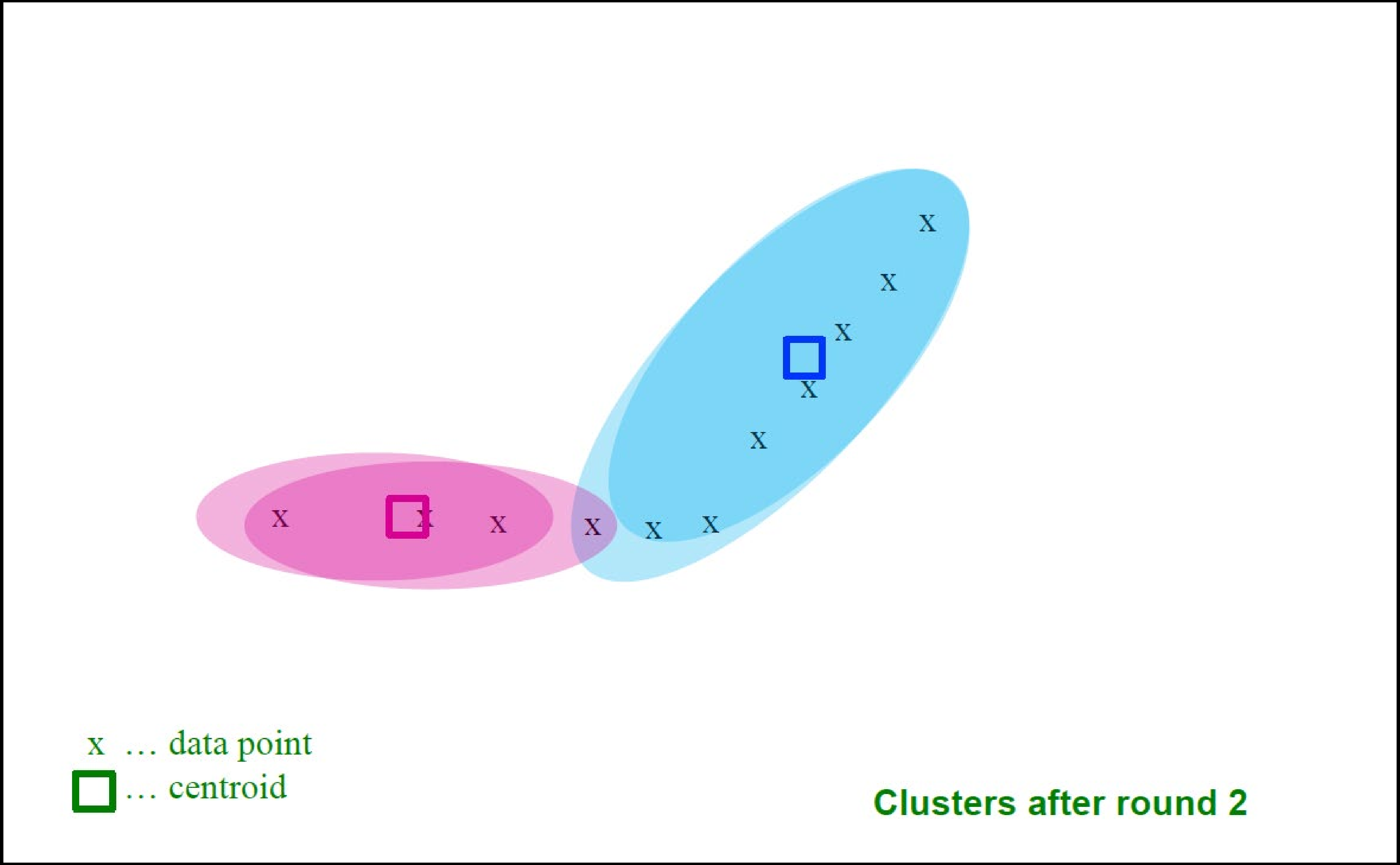
- Given data set $X = \{\mathbf{x}_1, \dots, \mathbf{x}_N\}$ and K clusters

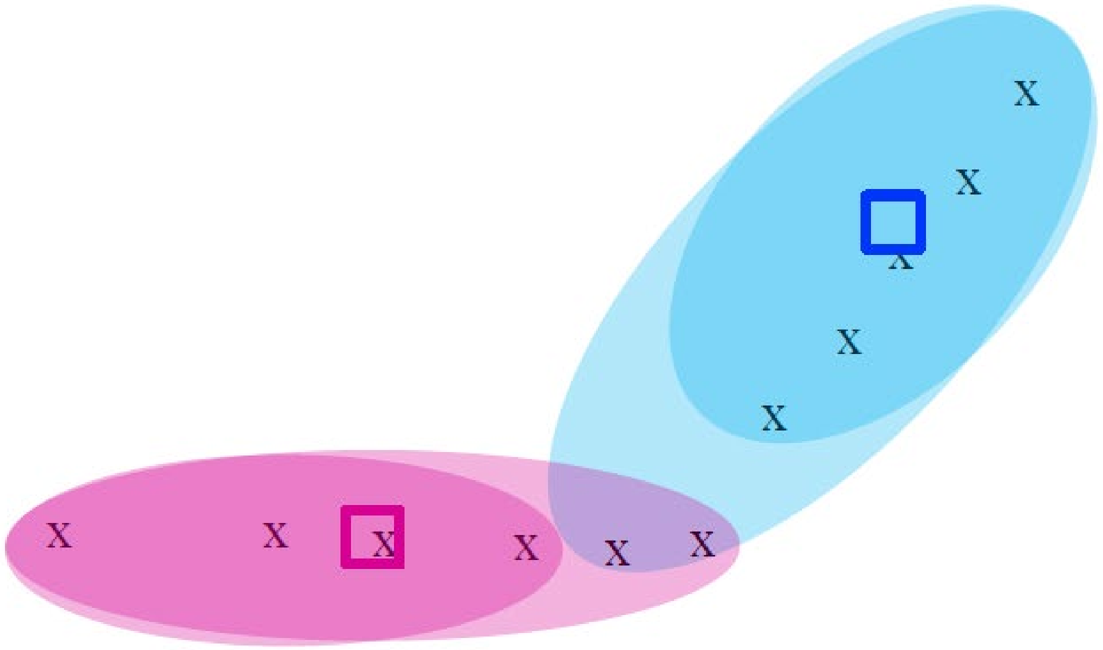
$$\min J = min \sum_{j=1}^{K} \sum_{i=1}^{N} \left\| \mathrm{x}_i - \mathbf{c}_j \right\|^2$$

where $\boldsymbol{c}_j$: centroid of the *j*-th cluster



Clusters after round 1


x … data point
… centroid
Clusters after round 2



x … data point

… centroid

Clusters at the end

Convergence?

- K-Means may not always converge to the right solution

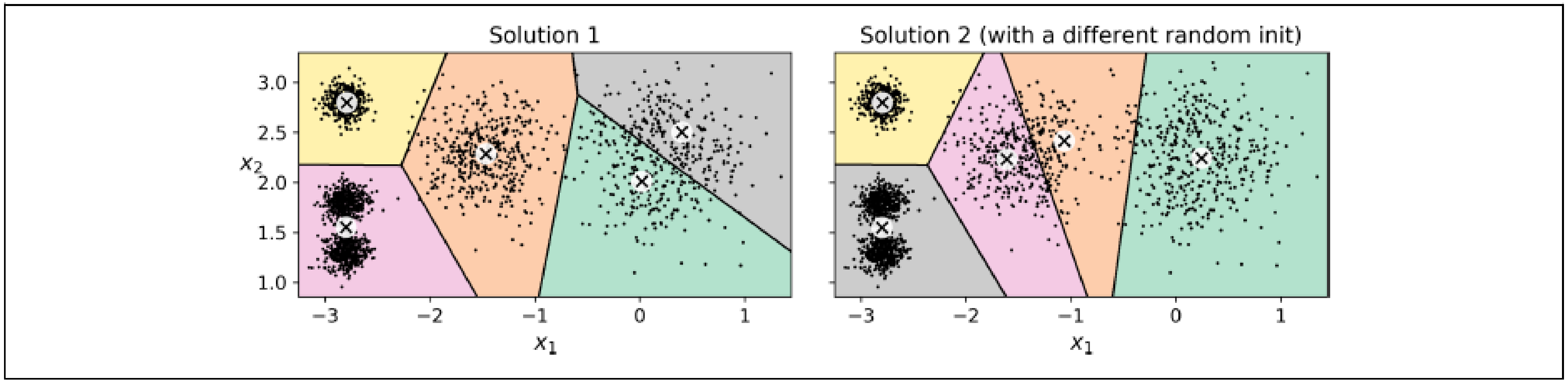
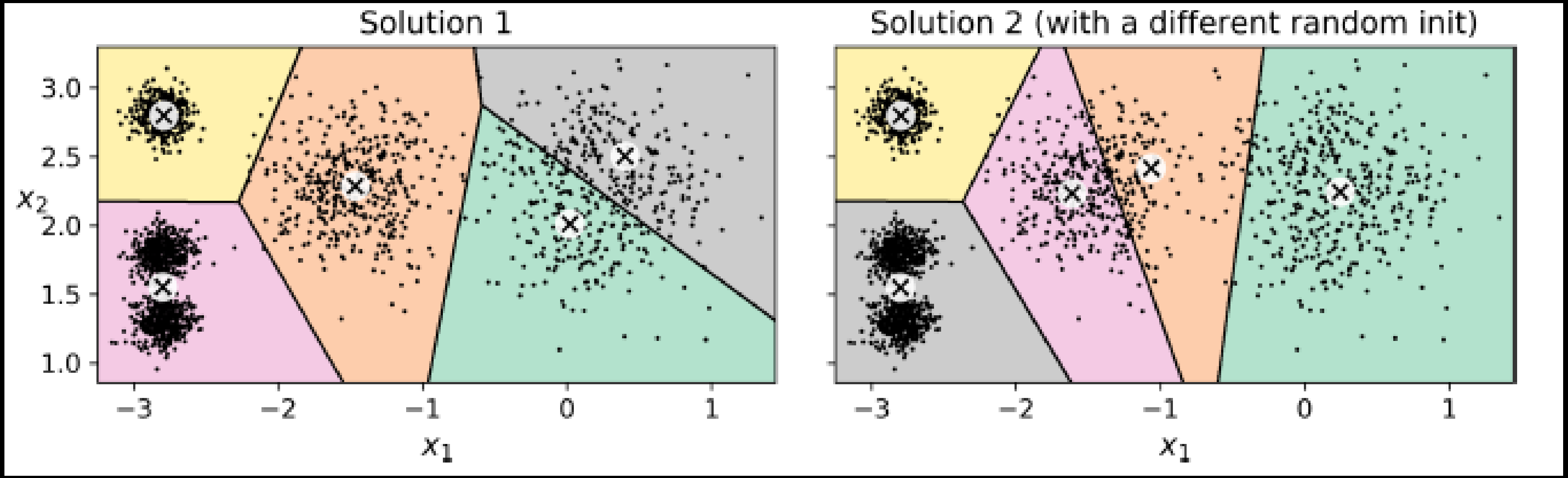


Figure 9-5. Sub-optimal solutions due to unlucky centroid initializations

Centroid Initialization Methods

- Use prior knowledge of where the centroids should be located
- Run algorithm multiple times with different random initializations
  - Use model's inertia (mean squared distance between each instance and its closest centroids) to find the best model
  - Solution 1 = 223.3
  - Solution 2 = 237.5



K++ Initialization Algorithm

- Randomly choose an instance from the dataset as a centroid
- Assign another instance from the dataset as another centroid with probability $\frac{D\left(\mathbf{x}^{(i)}\right)^2}{\sum_{j=1}^{m} D\left(\mathbf{x}^{(j)}\right)^2}$ where $D(\mathbf{x}^{(i)})$ is the distance between that instance and the closest centroid already chosen
- Repeat step 2 until all centroids are assigned

Variants of K-Means

- Accelerated K-Means
  - Uses the triangular inequality to keep track of the lower and upper bounds for distances between instances and centroids
- Mini-batch K-Means
  - Uses mini batches instead of the full dataset
  - Faster but has slightly worse inertia

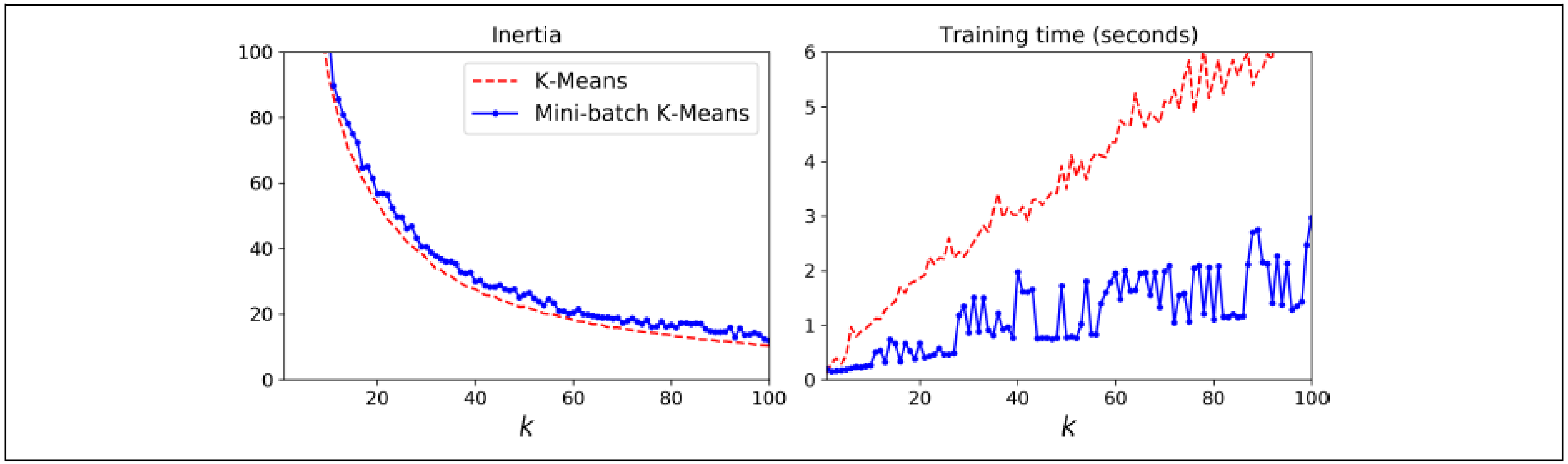


Figure 9-6. Mini-batch K-Means vs K-Means: worse inertia as k increases (left) but much faster (right)

Optimal Number of Clusters

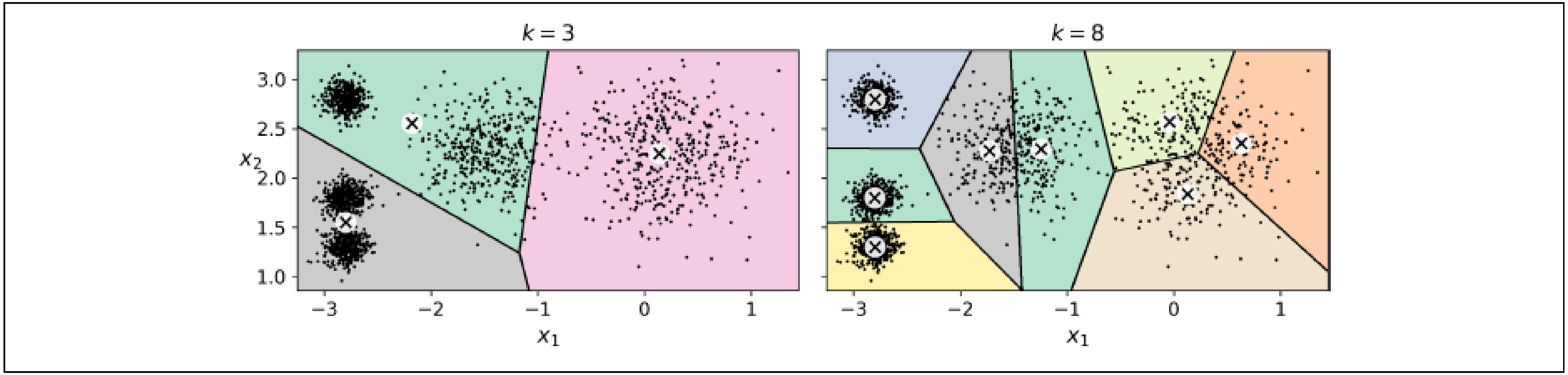


Figure 9-7. Bad choices for the number of clusters

- "Elbow" rule

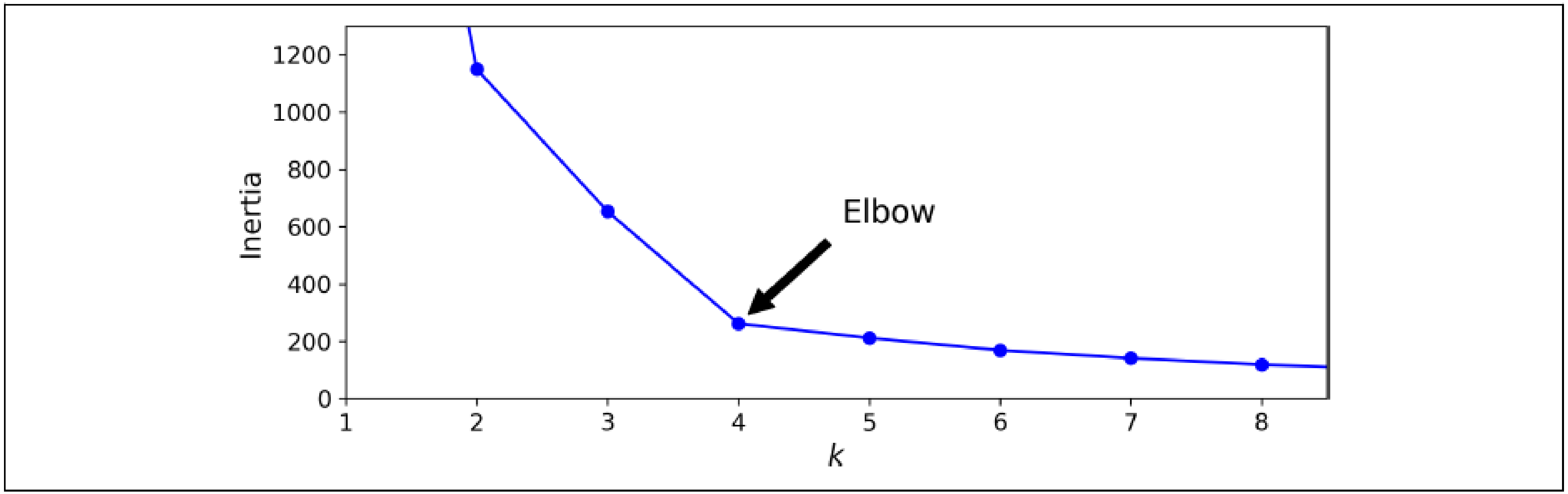


Figure 9-8. Selecting the number of clusters k using the "elbow rule"

- Silhouette Score
  - The mean silhouette coefficient over all instances
  - Silhouette coefficient = $(b-a)/\max(a,b)$ where a is the mean to other instances in the right cluster and b is the mean distance to the instances of the next closest cluster
  - Silhouette coefficient varies between -1 and +1
    - +1 : instance is well inside its own cluster
    - 0 : instance is close to the boundary
    - -1 : instance is in the wrong cluster

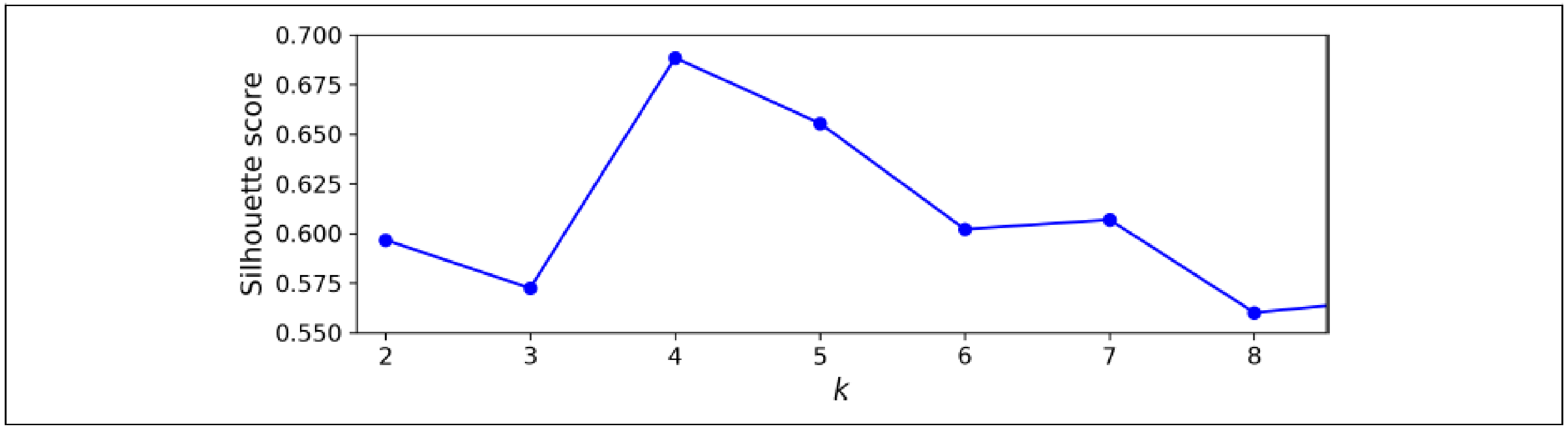


Figure 9-9. Selecting the number of clusters k using the silhouette score

- Silhouette Diagram

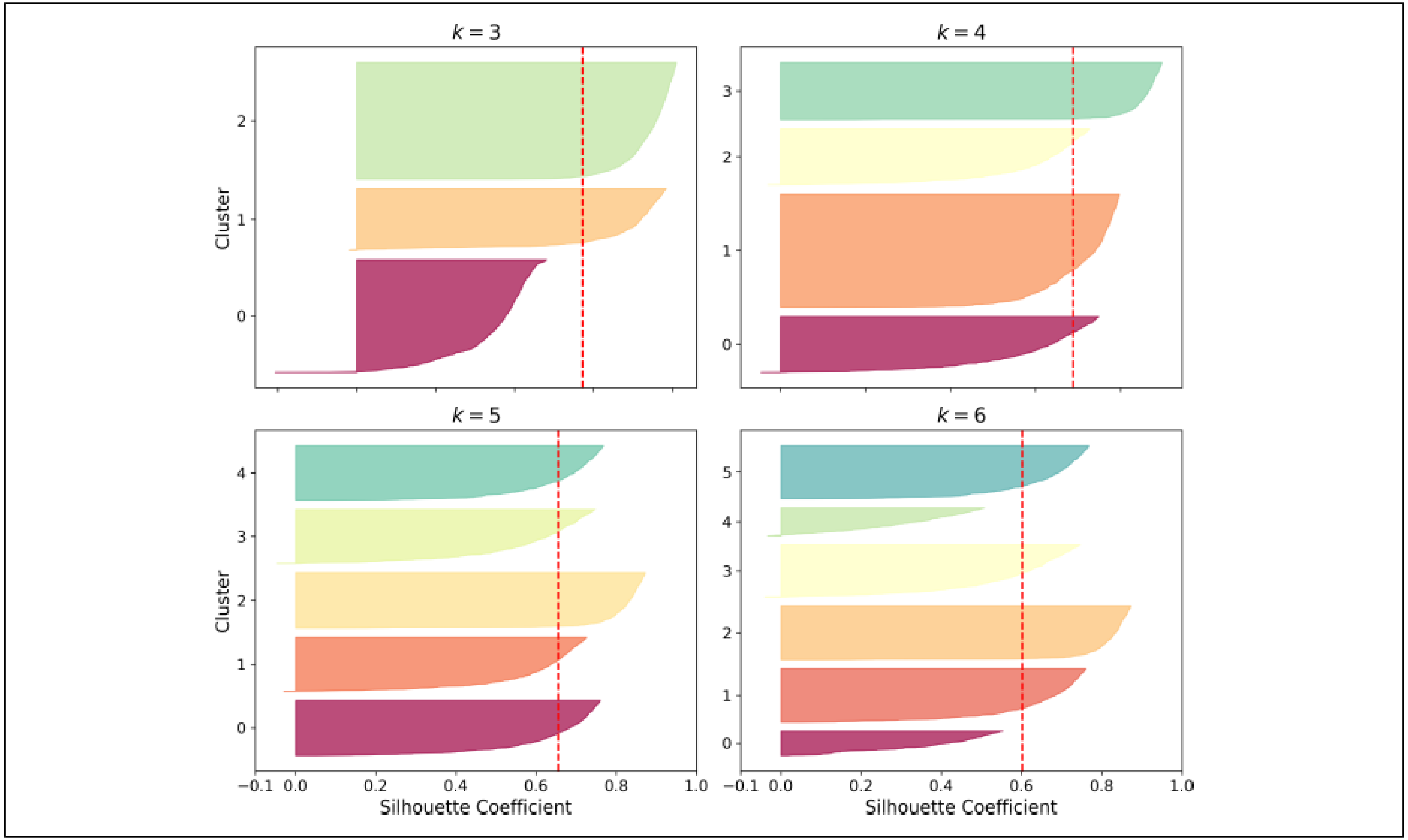
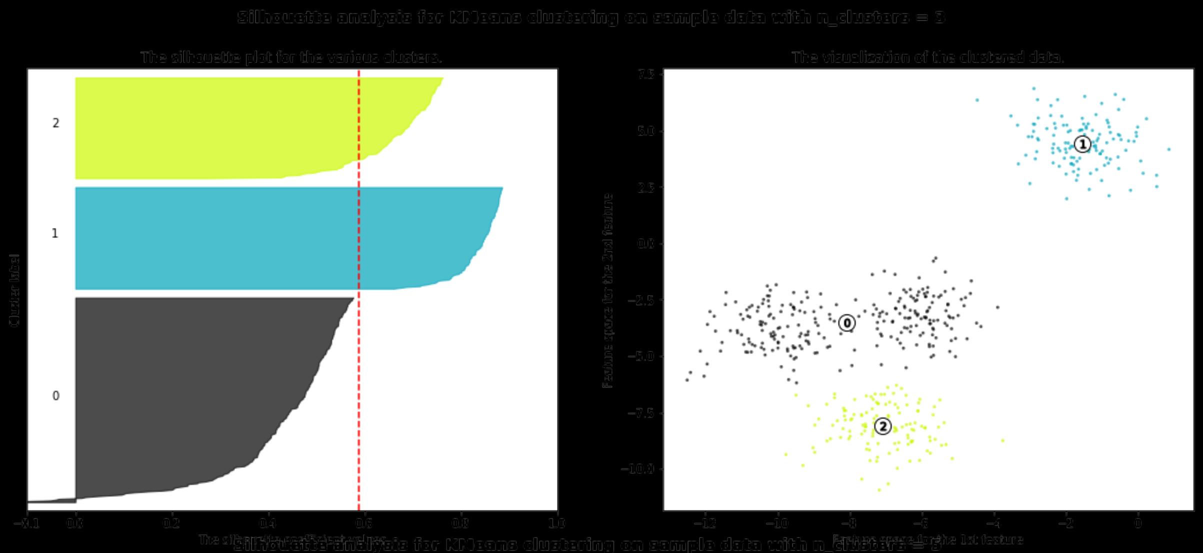
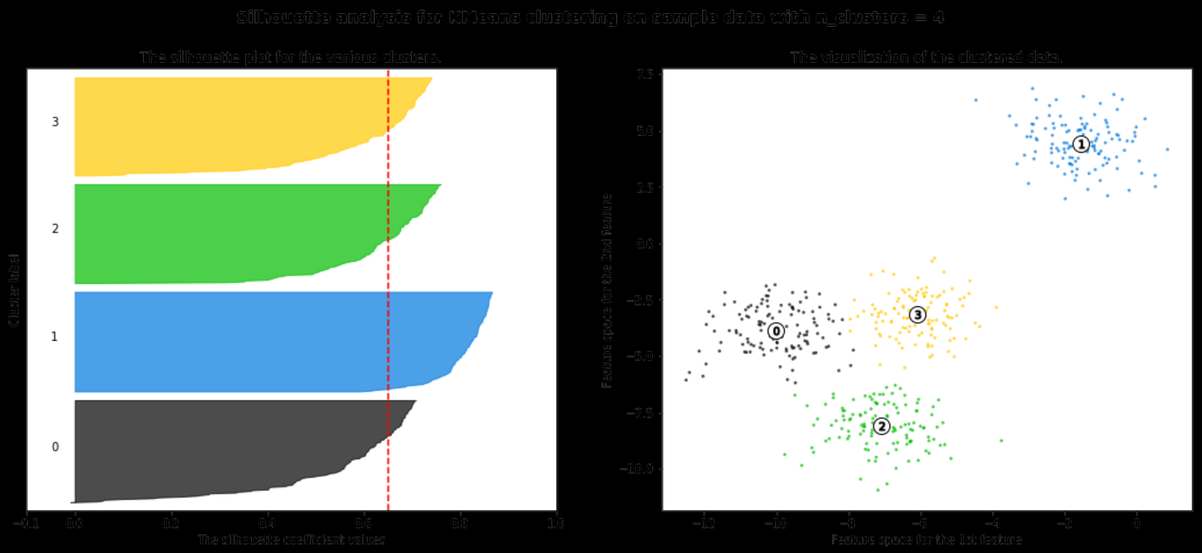
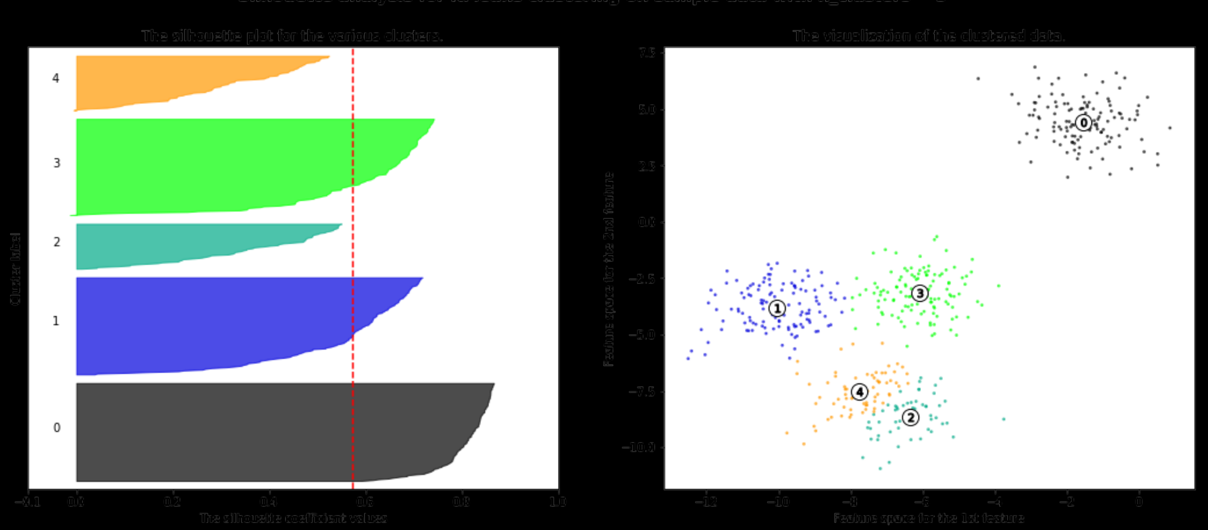


Figure 9-10. Silouhette analysis: comparing the silhouette diagrams for various values of k

- Silhouette Diagram for various values of *k*







- Choose *k*=4

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

Limits of K-Means

- Pros: popular, fast, scalable
- Cons: does not do well if clusters have different densities or non-spherical shapes (not compact and separable)

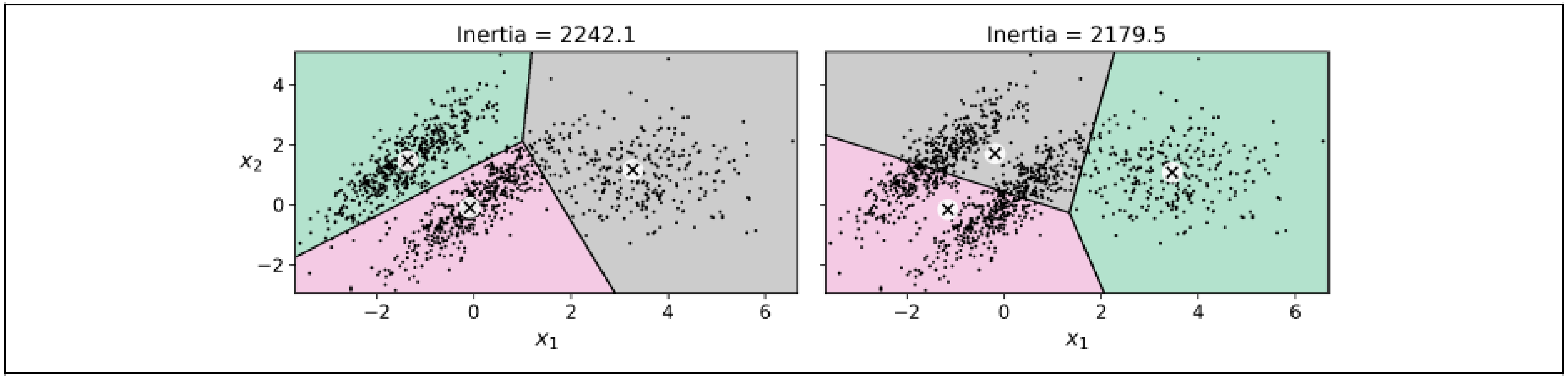


Figure 9-11. K-Means fails to cluster these ellipsoidal blobs properly

Clustering for Image Segmentation

- Partition an image into multiple segments
- Semantic segmentation assigns all pixels that are part of the same object type to the same segment
  - All pixels that are part of a pedestrian’s image are assigned to the *pedestrian* segment (i.e., one segment containing all the pedestrians)
- Instance segmentation assigns all pixels that are part of the same individual objects to the same segment
  - Each pedestrian has their own segment
- Color segmentation
  - Assigns pixels the same segment if they have similar color

Figure 9-12. Image segmentation using K-Means with various numbers of color clusters

Clustering for Preprocessing

- Consider the *digit* dataset (MNIST-like dataset 1,797 grayscale 8x8 images representing digits 0 to 9)
- Accuracy using Logistic Regression = 96.9 %
- Apply clustering with arbitrary number of clusters $k = 50$
  - Replace images with distances to these clusters
  - Accuracy = 97.8 %
- Find the best value of k using grid search ($k = 90$)
  - Accuracy = 98.2 %

Clustering for Semi-Supervised Learning

- Consider the digit dataset again
- Train Logistic Regression with only 50 instances
  - Accuracy = 83.3 %
- Cluster training set into 50 clusters, then find the 50 representative images (50 images closest to their respective centroids)

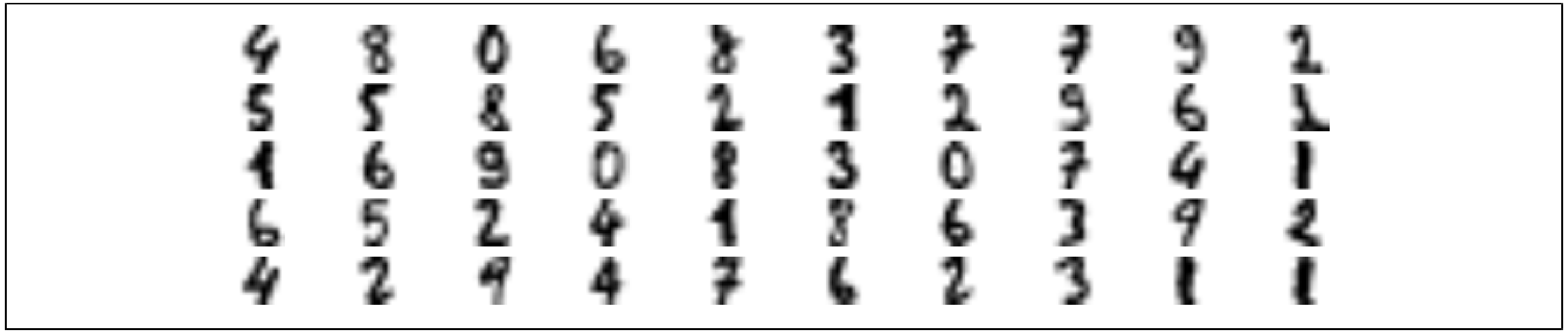


Figure 9-13. Fifty representative digit images (one per cluster)

- Manually label these images (4, 8, 0, …), i.e., still 50 labeled instances
  - Accuracy = 92.2 %

- Propagate the labels to all the other instances in the same cluster (*label propagation*)
  - Accuracy = 93.3 % (mislabeled instances close to the boundaries)
- Only propagate the labels to the 20% of the instances closest to the centroids
  - Accuracy = 94.0 % (vs 96.9 % using the full training set)

Active Learning

- Human expert interacts with the learning algorithm to provide labels as the algorithm needs them
- Uncertainty Sampling (most common strategy)
  - Train on available labeled instances
  - Use this model to make predictions on all unlabeled instances
  - Human expert labels the instances with low probability predictions
  - Iterate until acceptable performance is achieved

DBSCAN

- Create an ε-neighborhood that contains an instance with its neighbors located with a small distance ε from it
- If an instance has at least *min_samples* instances in its ε-neighborhood, then it becomes a core instance
- All instances in the neighborhood of a core instance belong to the same cluster
- If an instance is not a core instance and does not have one in its neighborhood, then it is an anomaly

- This algorithm works well for dense clusters that are well separated by low-density regions (e.g., the moons dataset)

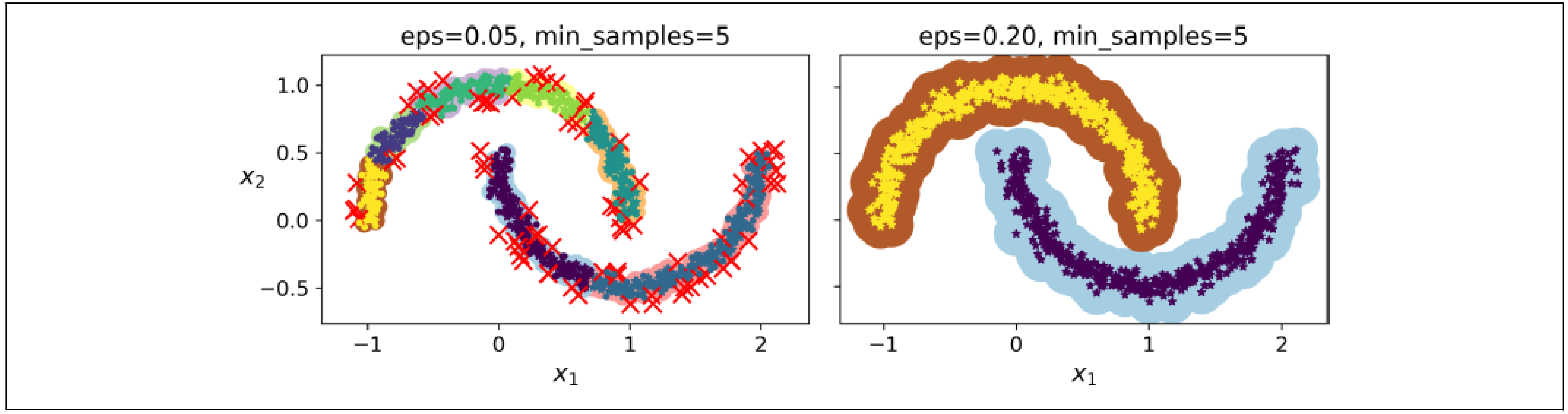


Figure 9-14. DBSCAN clustering using two different neighborhood radiuses

- The 4 instances are anomalies based on some maximum distance from the clusters

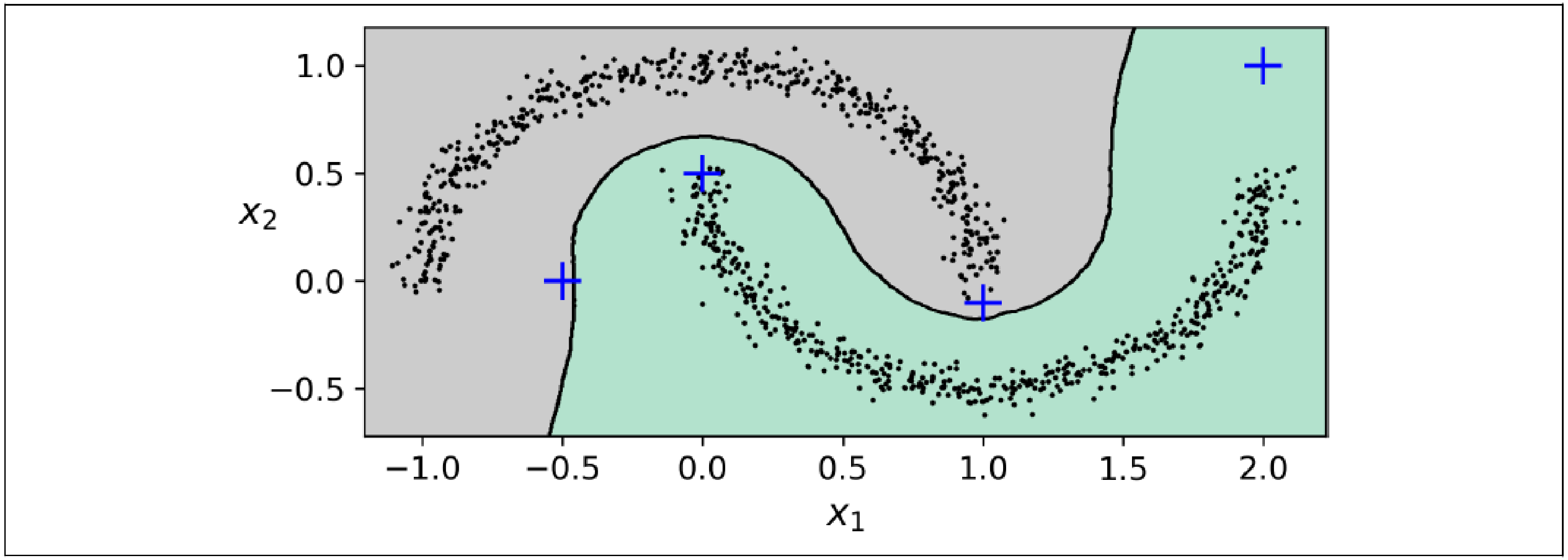
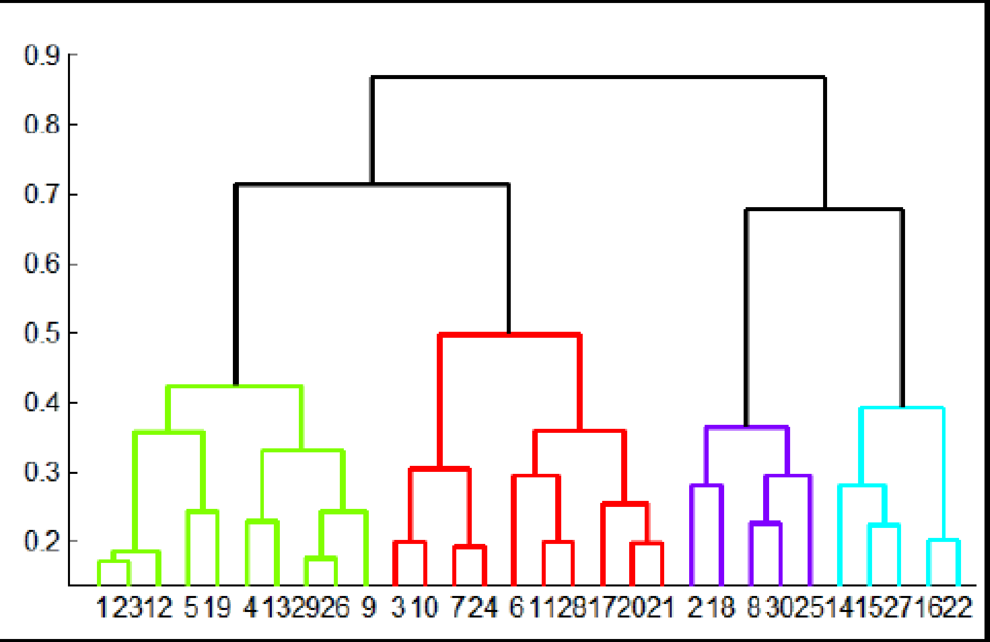


Figure 9-15. cluster_classification_diagram

Other Clustering Algorithms

- Agglomerative Clustering
- Mean-Shift
- Affinity Propagation
- Spectral Clustering

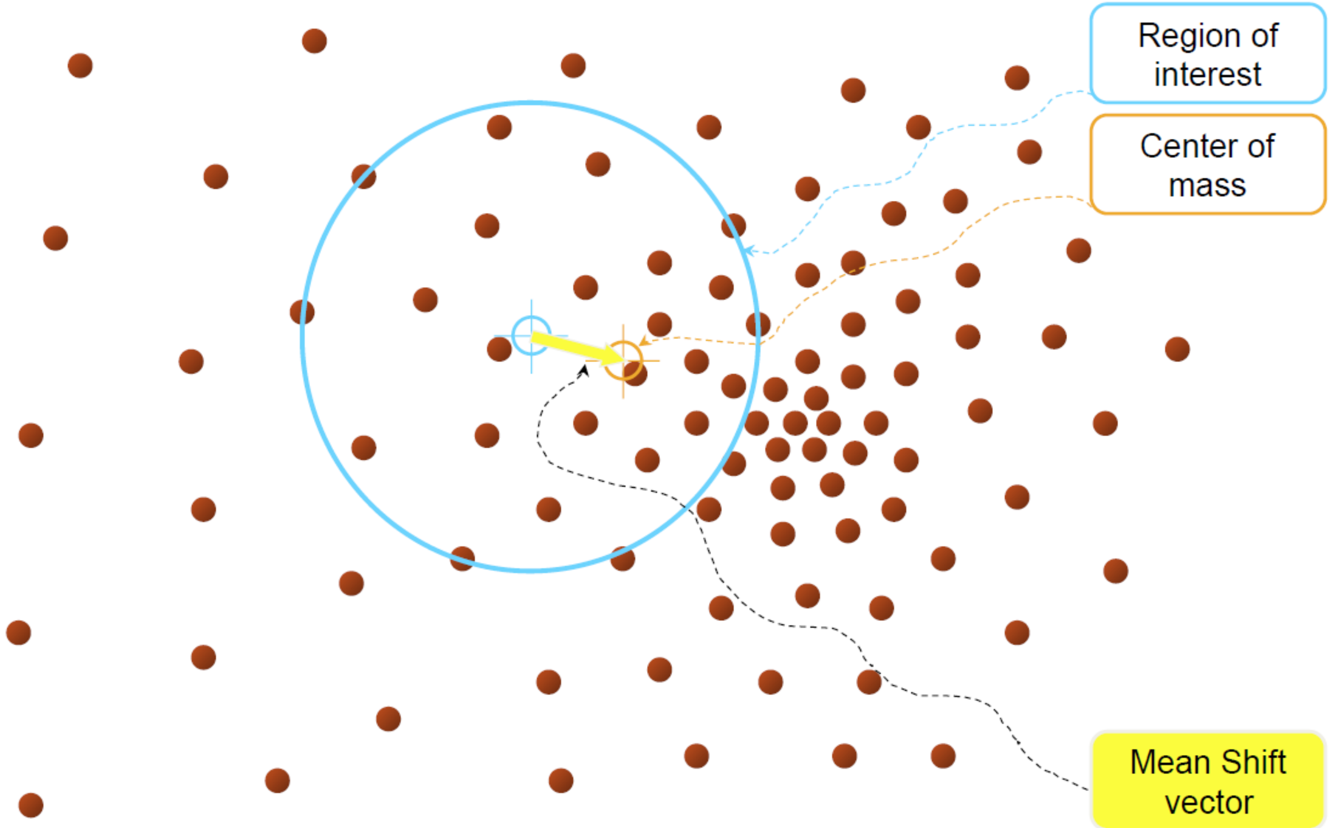
- Agglomerative Clustering
  - Bottom-up hierarchical clustering
  - Initially each point is a cluster
  - Repeated combine two nearest clusters into one



- Mean-shift
  - Place a circle then compute the mean of all instances within it
  - Shift the circle toward the mean
  - Example: K-Means Lecture by Fei-Fei Li

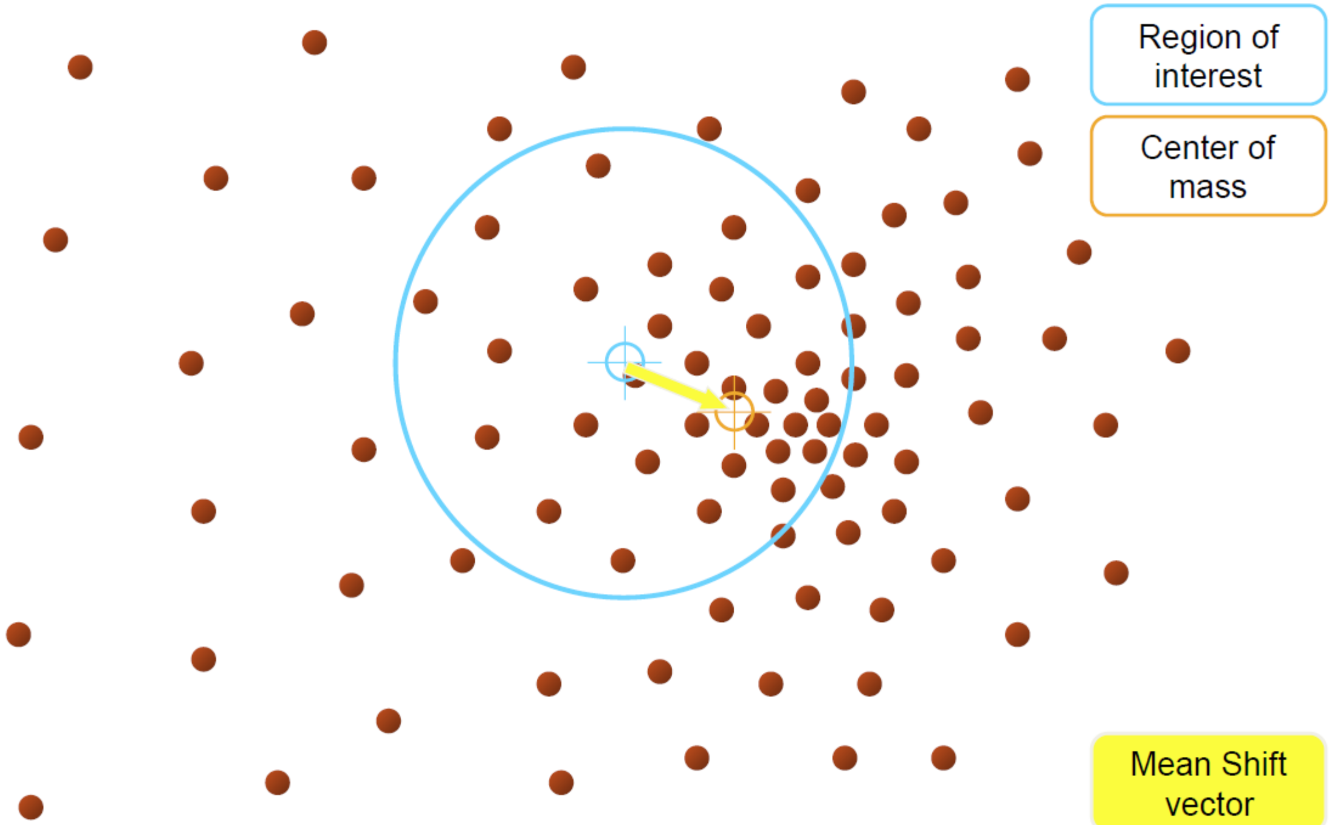
- Mean-shift
  - Place a circle then compute the mean of all instances within it
  - Shift the circle toward the mean
  - Example: K-Means Lecture by Fei-Fei Li

Mean-Shift



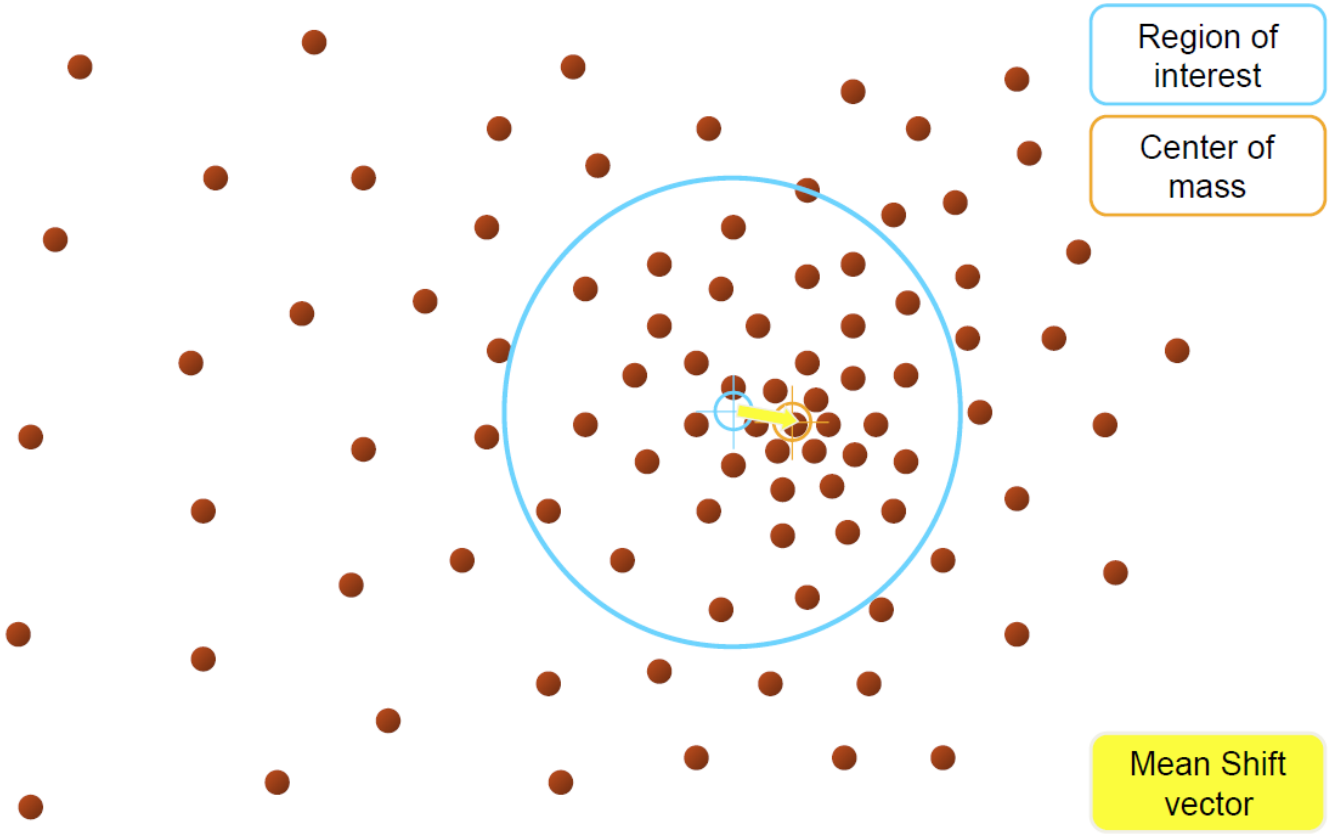
Slide by Y. Ukrainitz & B. Sarel

Mean-Shift



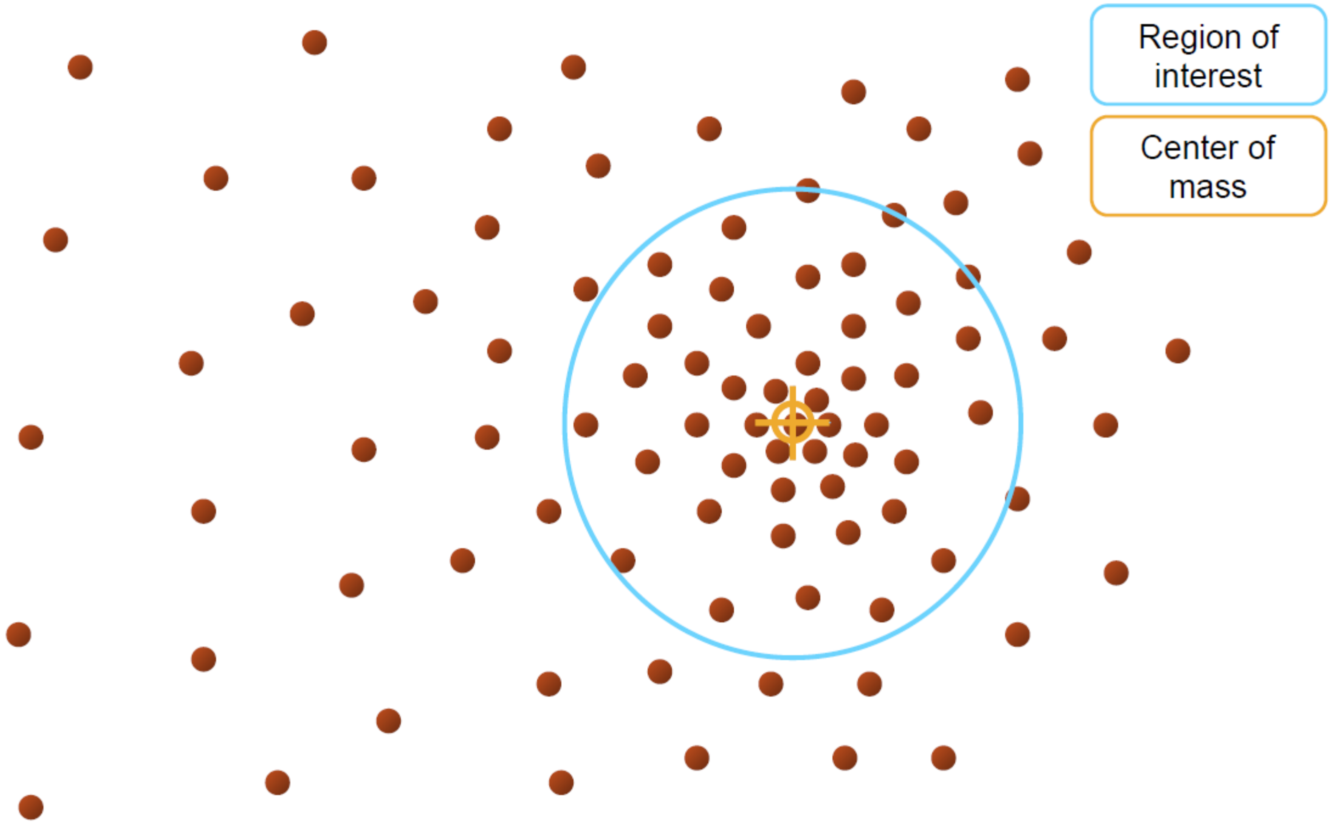
Slide by Y. Ukrainitz & B. Sarel

Mean-Shift



Slide by Y. Ukrainitz & B. Sarel

Mean-Shift

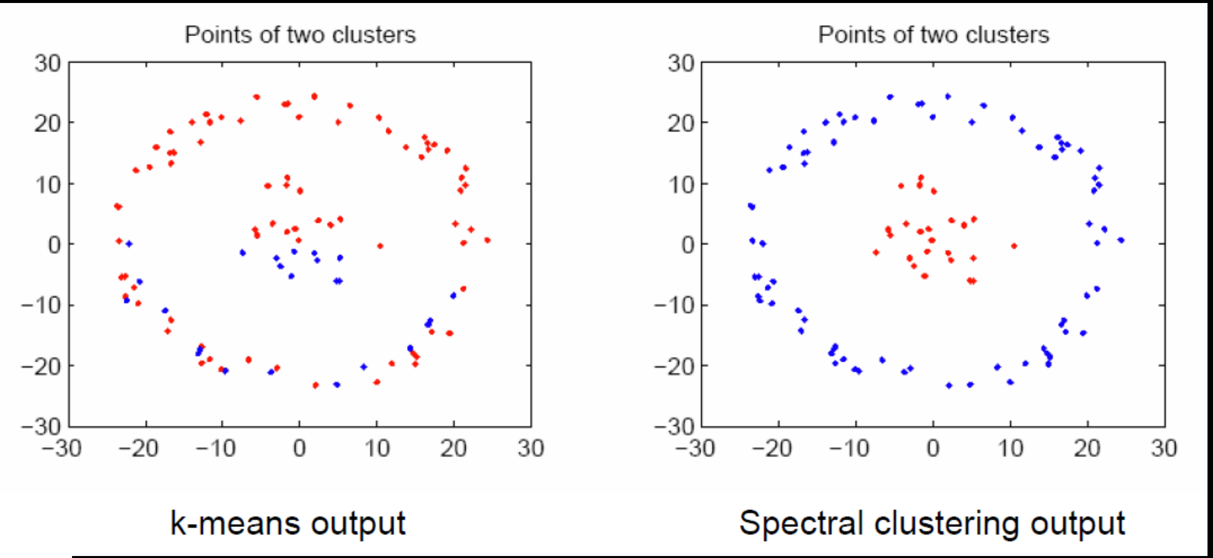


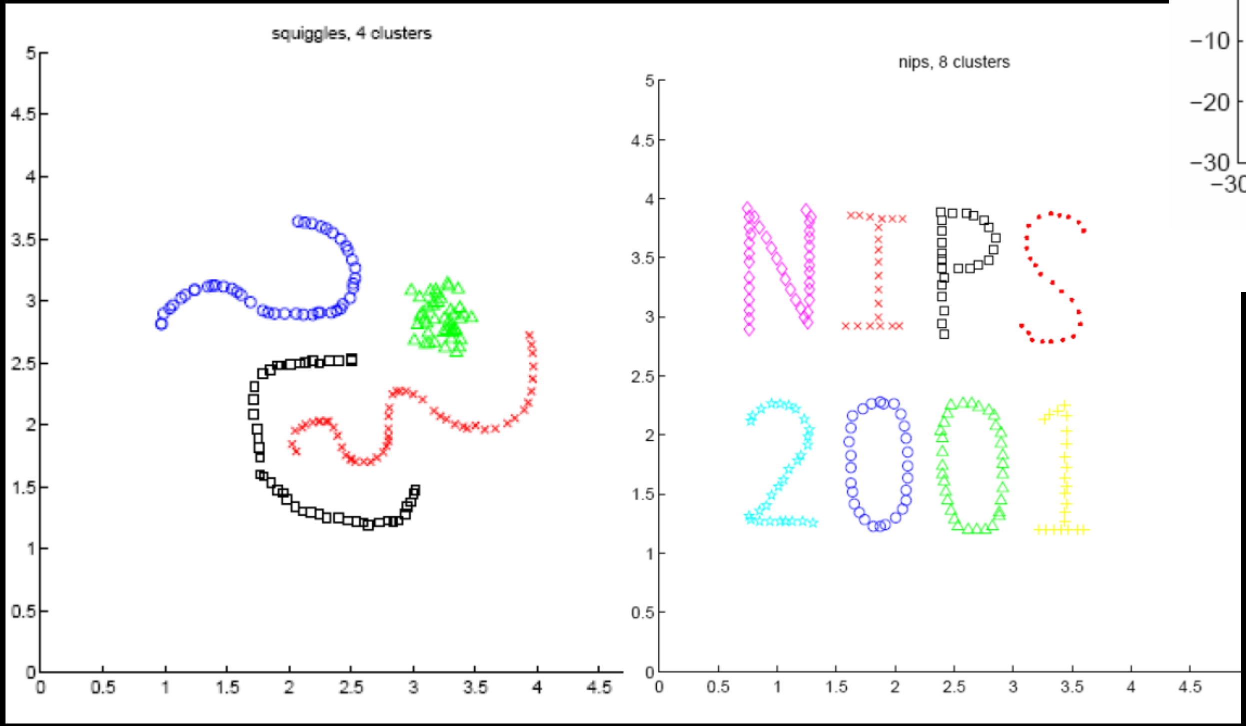
Slide by Y. Ukrainitz & B. Sarel

- Affinity Propagation
  - Instances vote for similar instances to be their representatives
  - Once the algorithm converges, each representative and its voters form a cluster

  - Example: Clustering Algorithms - From Start To Start of The Art by Lovro Iliassich
  - https://uploads.toptal.io/blog/image/92526/toptal-blog-image-1463639329606-7297e0c0f8be49f7f9105830d76848ea.gif

- Spectral Clustering
  - Create a similarity graph between instances
  - Create Laplacian matrix and use the first *k* eigenvectors to define feature vectors
  - Run K-Means on the feature vectors

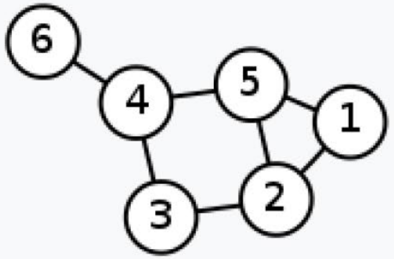




- Capable of creating complex cluster structures
- Does not scale well to large datasets

Laplacian Matrix

- Laplacian matrix is a matrix representation of a graph
- L = D – A

Labelled graph	Degree matrix	Adjacency matrix	Laplacian matrix
	$\begin{pmatrix} 2 & 0 & 0 & 0 & 0 & 0 \\ 0 & 3 & 0 & 0 & 0 & 0 \\ 0 & 0 & 2 & 0 & 0 & 0 \\ 0 & 0 & 0 & 3 & 0 & 0 \\ 0 & 0 & 0 & 0 & 3 & 0 \\ 0 & 0 & 0 & 0 & 0 & 1 \end{pmatrix}$	$\begin{pmatrix} 0 & 1 & 0 & 0 & 1 & 0 \\ 1 & 0 & 1 & 0 & 1 & 0 \\ 0 & 1 & 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 & 1 & 1 \\ 1 & 1 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 & 0 \end{pmatrix}$	$\begin{pmatrix} 2 & -1 & 0 & 0 & -1 & 0 \\ -1 & 3 & -1 & 0 & -1 & 0 \\ 0 & -1 & 2 & -1 & 0 & 0 \\ 0 & 0 & -1 & 3 & -1 & -1 \\ -1 & -1 & 0 & -1 & 3 & 0 \\ 0 & 0 & 0 & -1 & 0 & 1 \end{pmatrix}$

- Example of a labeled undirected graph from Wikipedia