Clustering Algorithms

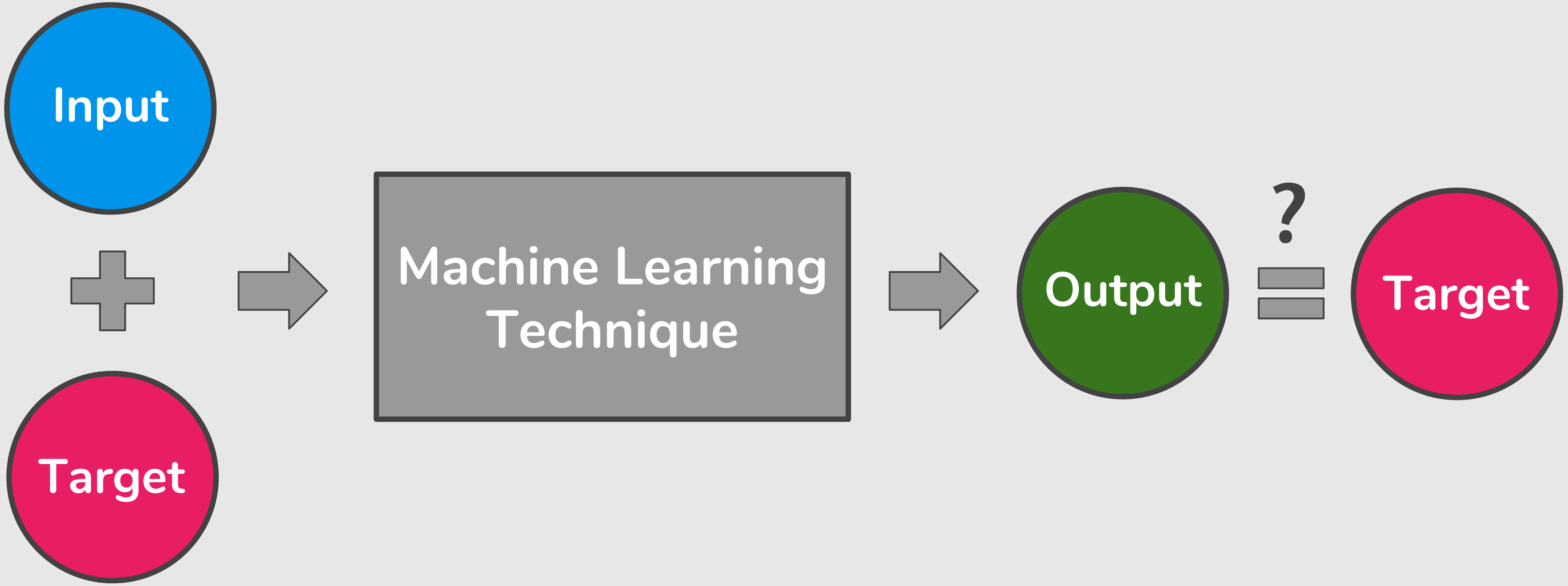
Machine Learning and Pattern Recognition

Prof. Sandra Avila

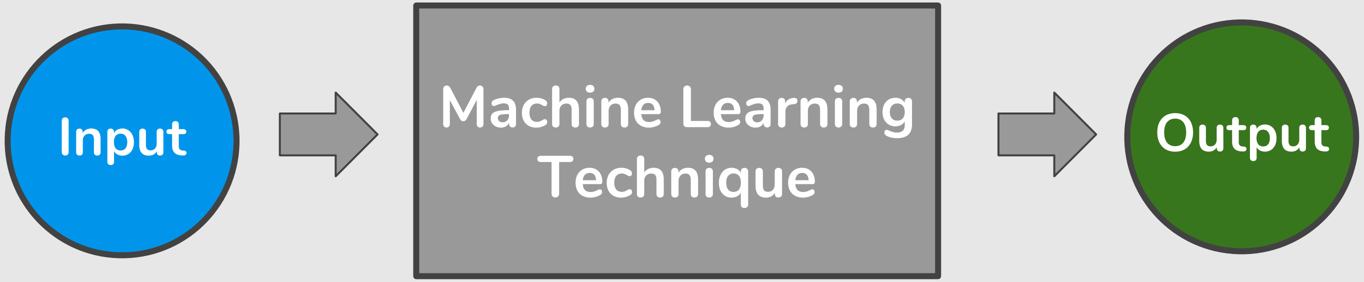
Institute of Computing (IC/Unicamp)

MC886/MO444, September 20, 2018

Supervised Learning



Unsupervised Learning



The goal of unsupervised learning is **to find patterns** in the data, and build new and useful representations of it.

Clustering

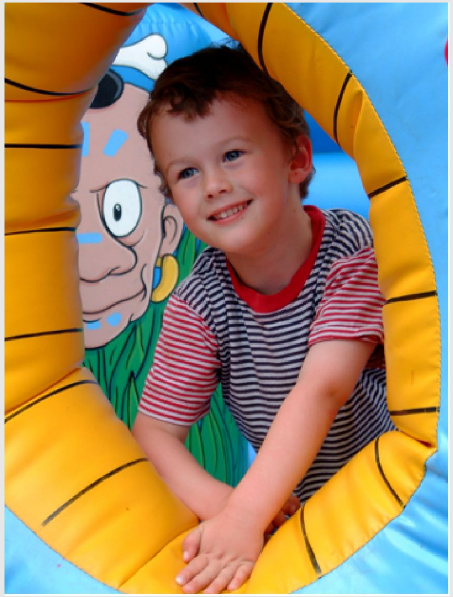
k-Means Algorithm

Did anyone say pizza?

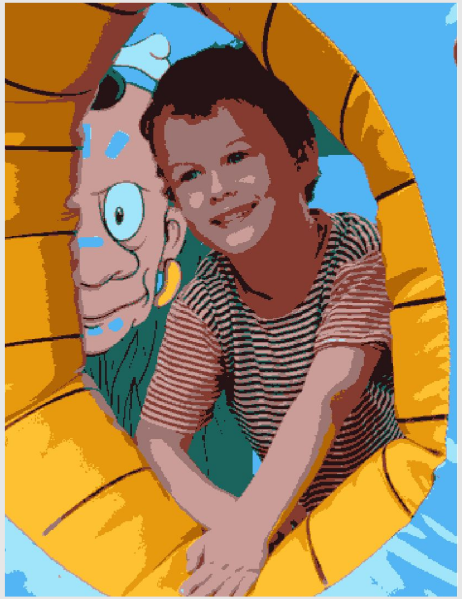
Praça do Ciclo Básico
Instituto de Artes
Instituto de Filosofia e Ciências Humanas
IC - Institute of Computing
EXTECAMP - Escola de Extensão da Unicamp
IA - Instituto de Artes
Ginásio - fac Educação Fisica Unicamp
Galeria de Arte Unicamp
Restaurante Universitário - Unicamp
Instituto de Física Gleb Wataghin (IFGW...
Fonoteca Neotropical Jacques Vielliard (FNJV)
BIQ - Biblioteca do Instituto de Química
7° Distrito Policial de Campinas
Metropoly Bar
UNICAMP - Faculdade de Engenharia Química
Faculdade de Engenharia Elétrica e de Computação
Park Praça da Paz
Auto Repair Shop Posto Ipiranga
Av. Albert Einstein
Av. Professor Atílio Martini
R. Pitágoras
R. Lev Landau
R. Sérgio Buarque de Holanda
R. Roxo Moreira
R. Dr. Shigeo Mori
Google

k-Means: Image Segmentation

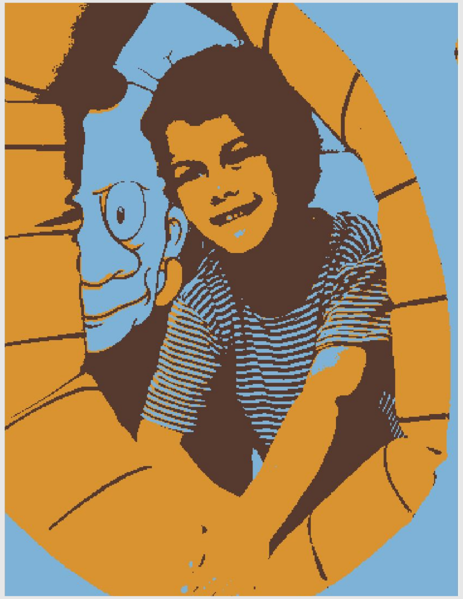
Original



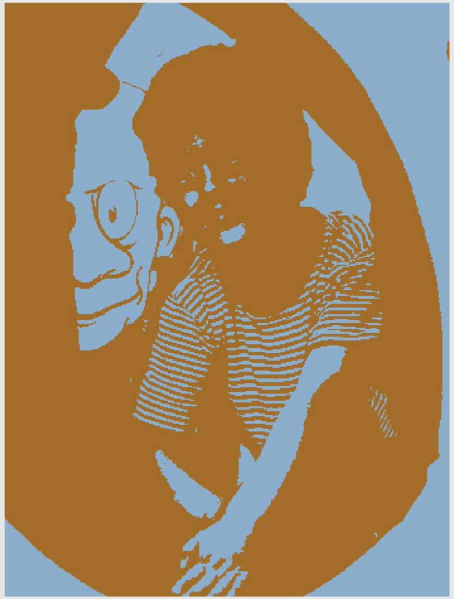
K = 10



K = 3



K = 2



Credit: Christopher Bishop

k-Means Algorithm

1. Define the *k* centroids.
2. Find the closest centroid & update cluster assignments.
3. Move the centroids to the center of their clusters.
4. Repeat steps 2 and 3 until the centroid stop moving a lot at each iteration (i.e., until the algorithm converges).

k-Means Algorithm

$$\min_k ||x^{(i)} - \mu_k||$$

```
Randomly initialize $K$ cluster centroids $\mu_1, \mu_2, \ldots, \mu_K \in \mathbb{R}^n$
repeat {
    for $i$ = 1 to $m$
        $c^{(i)}$ := index (from 1 to $K$) of cluster centroid **closest** to $x^{(i)}$
    for $k$ = 1 to $K$
        $\mu_k$ := mean of points assigned to cluster $k$
}
```

Clustering

Optimization Objective

k-Means Optimization Objective

$c^{(i)}$ = index of cluster (from 1 to K) to which example $x^{(i)}$ is currently assigned

μ_k = cluster centroid k

$\mu_{c^{(i)}}$ = cluster centroid of cluster to which example $x^{(i)}$ has been assigned

k-Means Optimization Objective

$c^{(i)}$ = index of cluster (from 1 to K) to which example $x^{(i)}$ is currently assigned

μ_k = cluster centroid k

$\mu_{c^{(i)}}$ = cluster centroid of cluster to which example $x^{(i)}$ has been assigned

Optimization objective:

$$J(c^{(1)}, \dots, c^{(m)}, \mu_1, \dots, \mu_K) = \frac{1}{m} \sum_{i=1}^{m} \|x^{(i)} - \mu_{c^{(i)}}\|$$

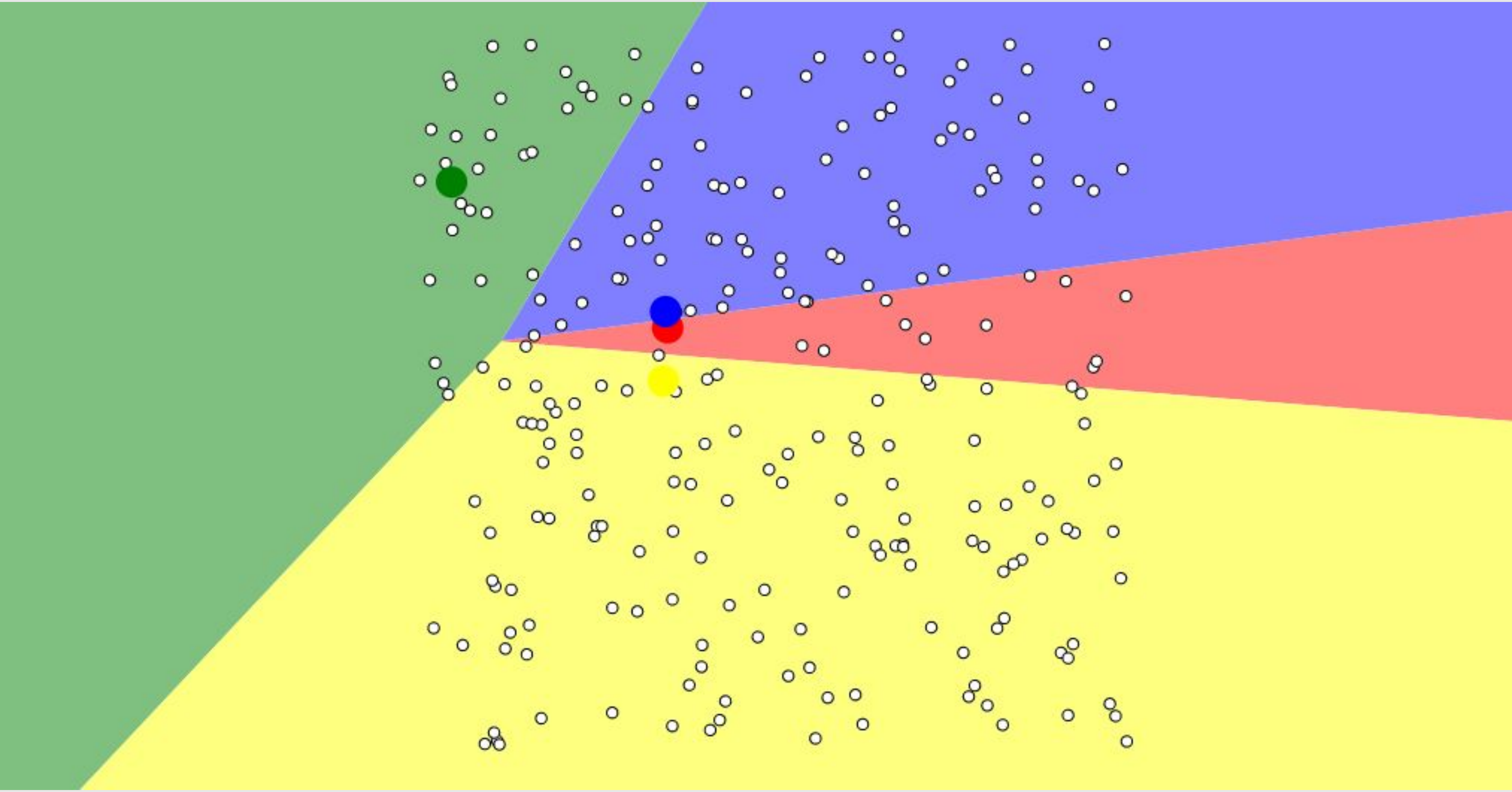
$$\min_{\substack{c^{(1)}, \dots, c^{(m)} \\ \mu_1, \dots, \mu_K}} J(c^{(1)}, \dots, c^{(m)}, \mu_1, \dots, \mu_K)$$

k-Means Optimization Objective

```
Randomly initialize $K$ cluster centroids $\mu_1, \mu_2, \ldots, \mu_K \in \mathbb{R}^n$
repeat {
    for $i$ = 1 to $m$
        $c^{(i)}$ := index (from 1 to $K$) of cluster centroid **closest** to $x^{(i)}$
    for $k$ = 1 to $K$
        $\mu_k$ := mean of points assigned to cluster $k$
}
```

Clustering

Random Initialization



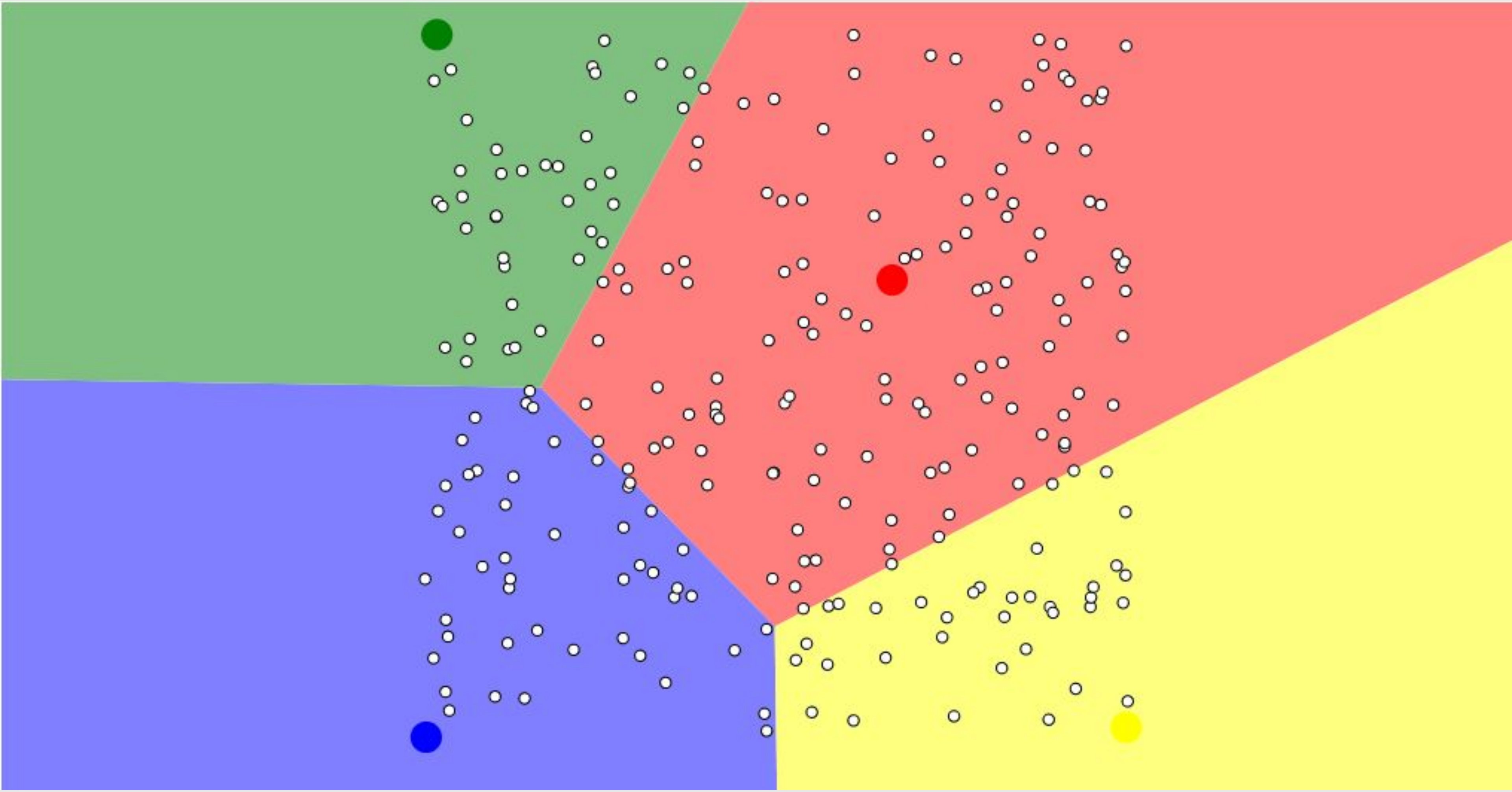
Random Initialization

```
for $i$ = 1 to 100 {
    Randomly initialize k-Means.
    Run k-Means. Get $c^{(1)}, \ldots, c^{(m)}, \mu_1, \ldots, \mu_K$.
    Compute cost function $J$.
}
```

Pick clustering that gave lowest cost $J(c^{(1)}, \ldots, c^{(m)}, \mu_1, \ldots, \mu_K)$.

Can we do better?

- One idea for initializing k-Means is to use a farthest-first traversal on the data set, **to pick K points that are far away from each other**.



Can we do better?

- One idea for initializing k-Means is to use a farthest-first traversal on the data set, to pick K points that are far away from each other.

- However, this is **too sensitive to outliers**.

k-Means++ (Arthur & Vassilvitski, 2007)

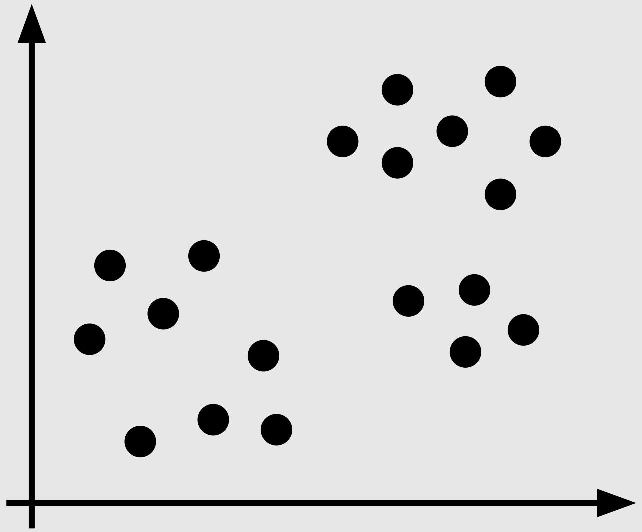
- It works similarly to the “farthest” heuristic.
- Choose each point at random, with probability proportional to its squared distance from the centers chosen already.

scikit-learn (default)

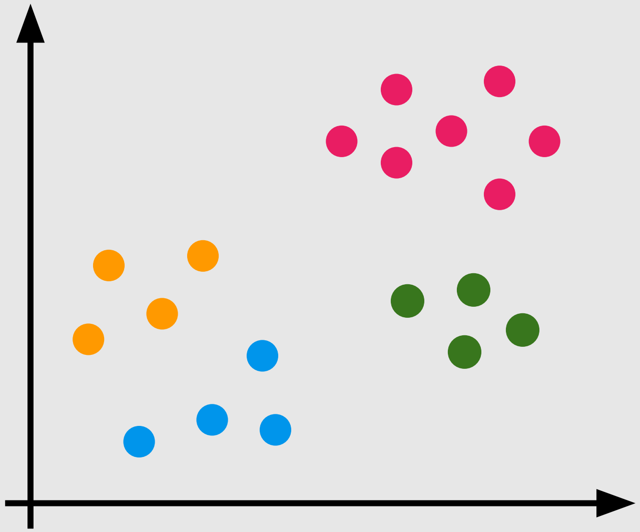
Clustering

Choosing the number of clusters

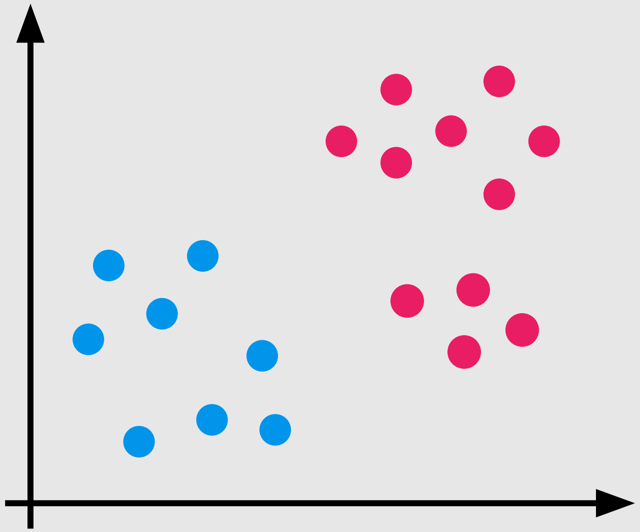
What is the right value of K?



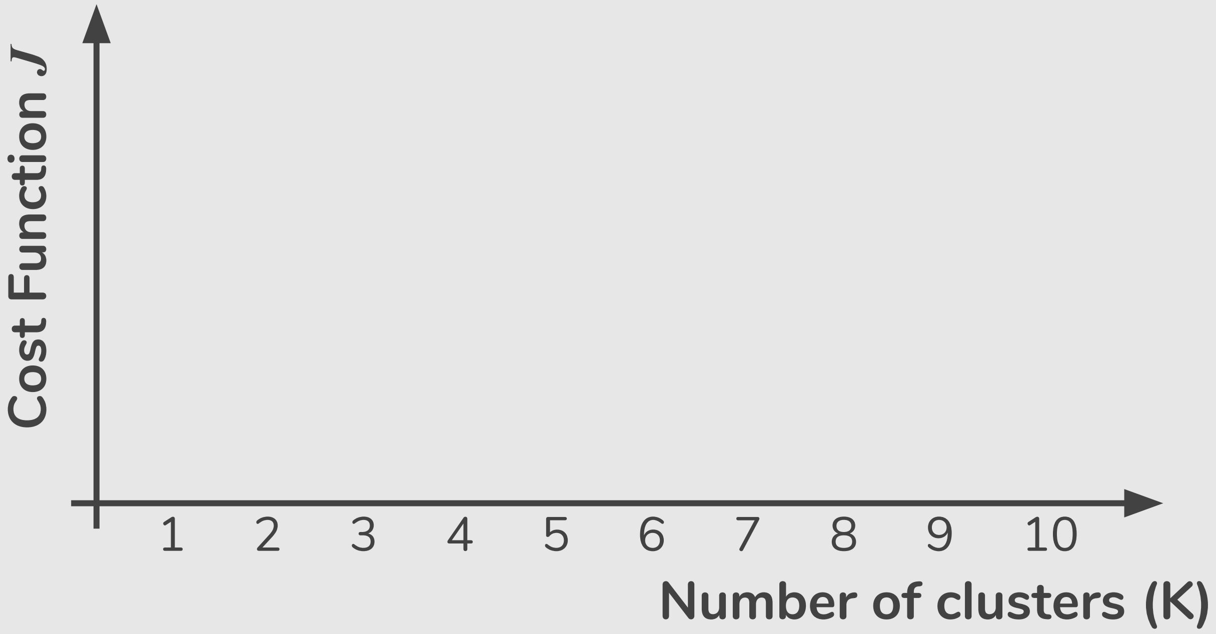
What is the right value of K?



What is the right value of K?

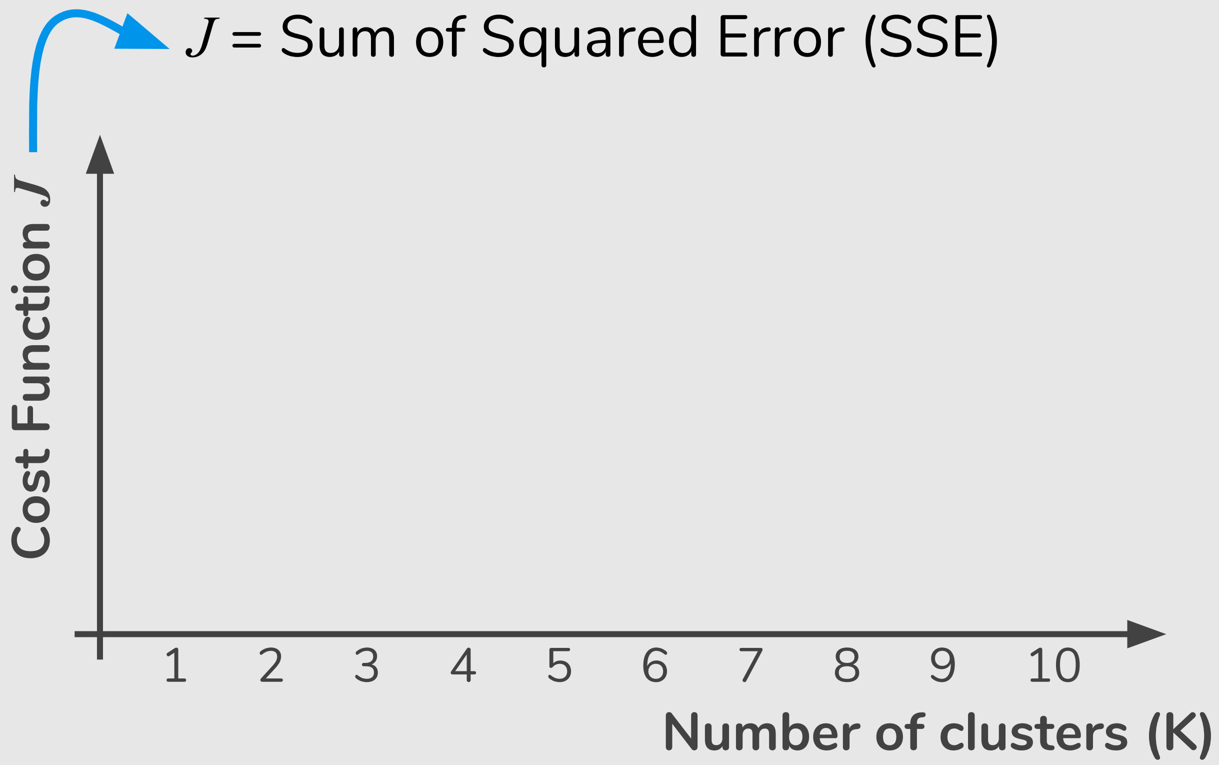


Elbow Method



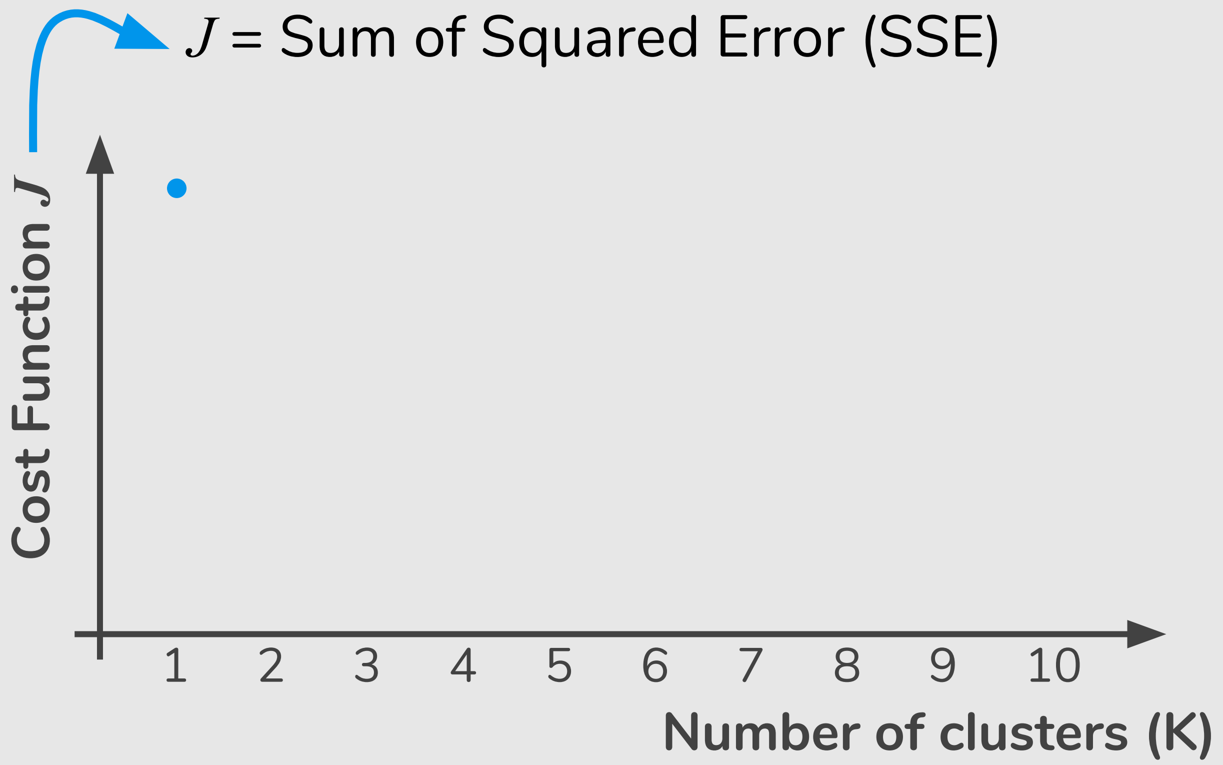
Elbow Method

J = Sum of Squared Error (SSE)



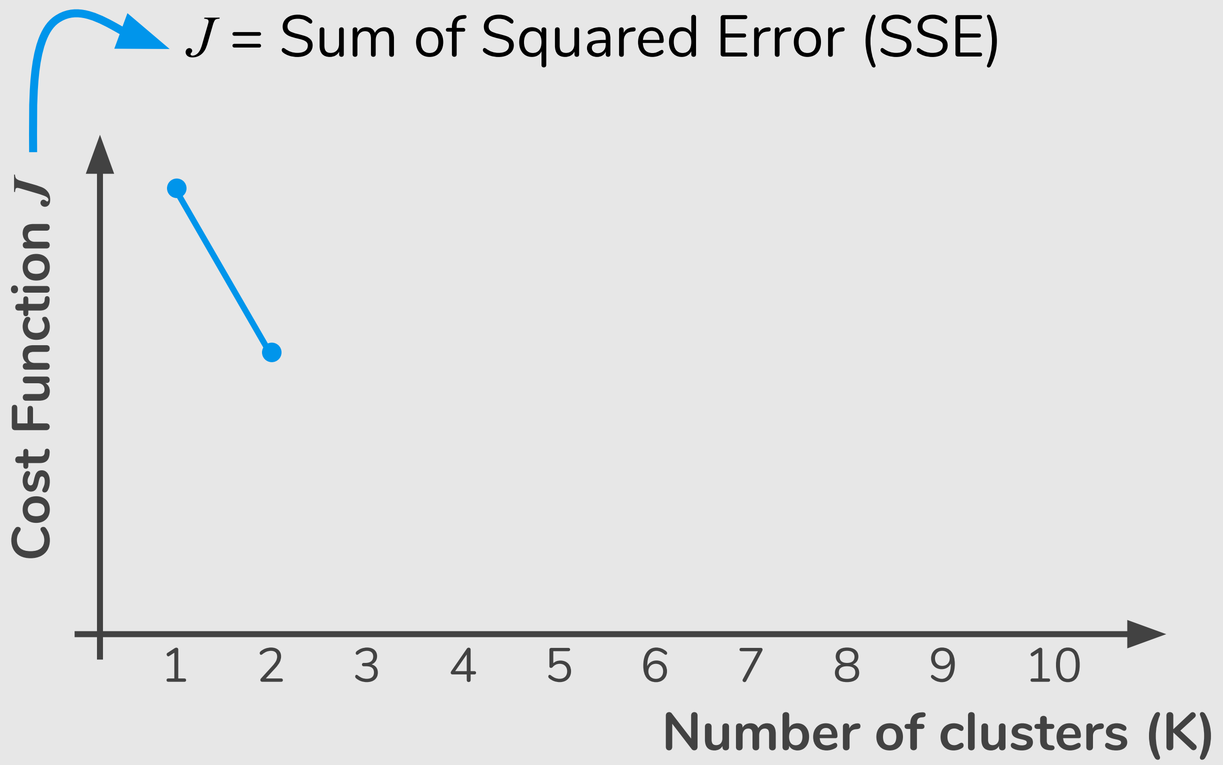
Elbow Method

J = Sum of Squared Error (SSE)



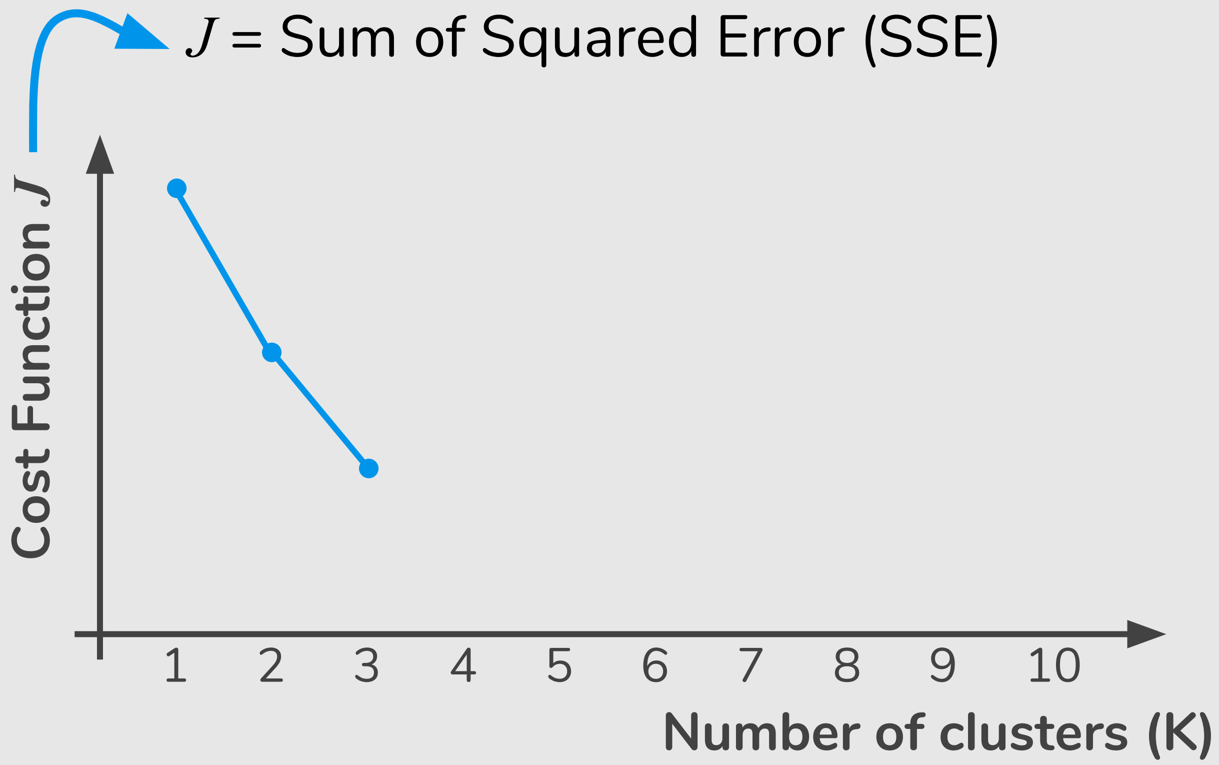
Elbow Method

J = Sum of Squared Error (SSE)



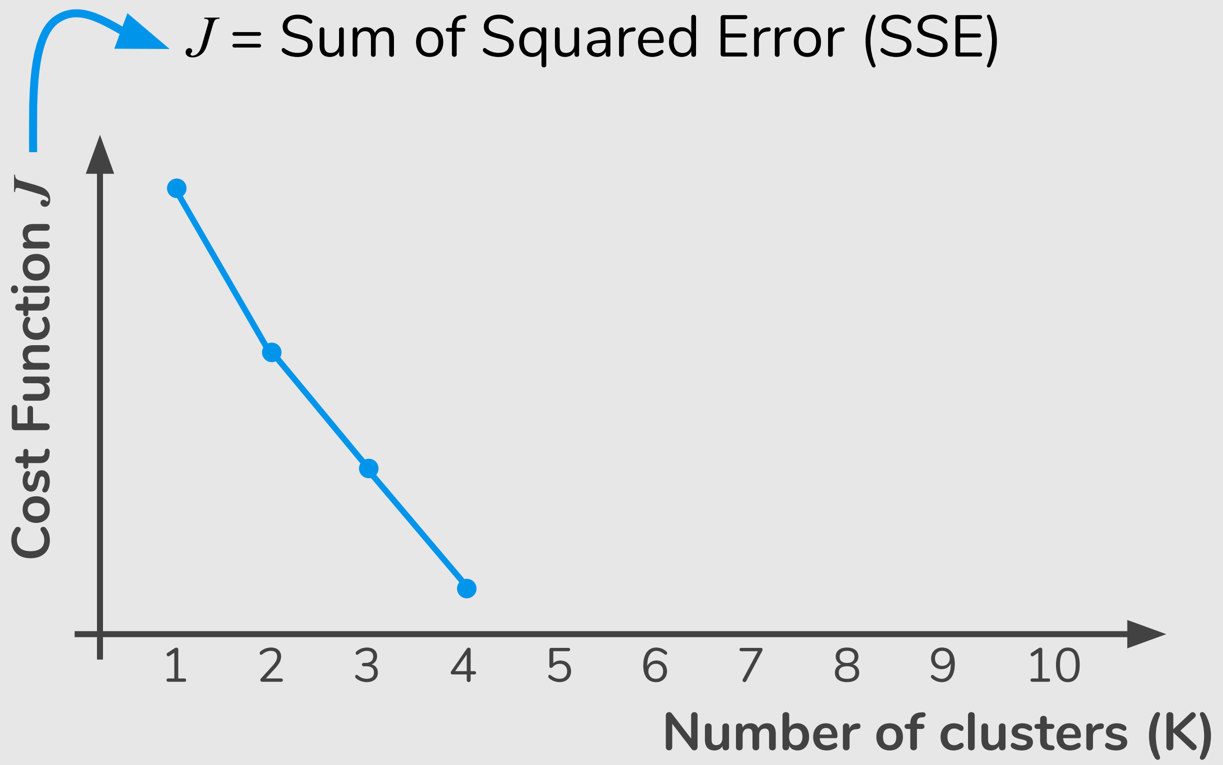
Elbow Method

J = Sum of Squared Error (SSE)



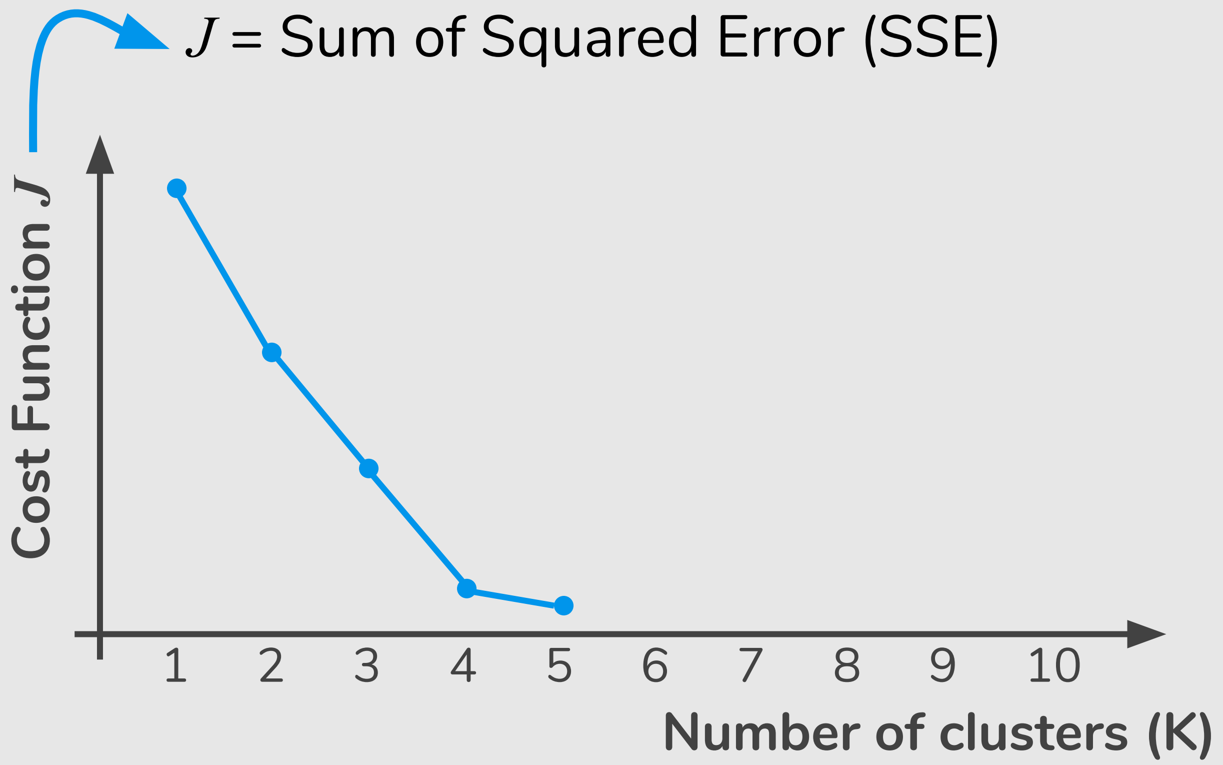
Elbow Method

J = Sum of Squared Error (SSE)



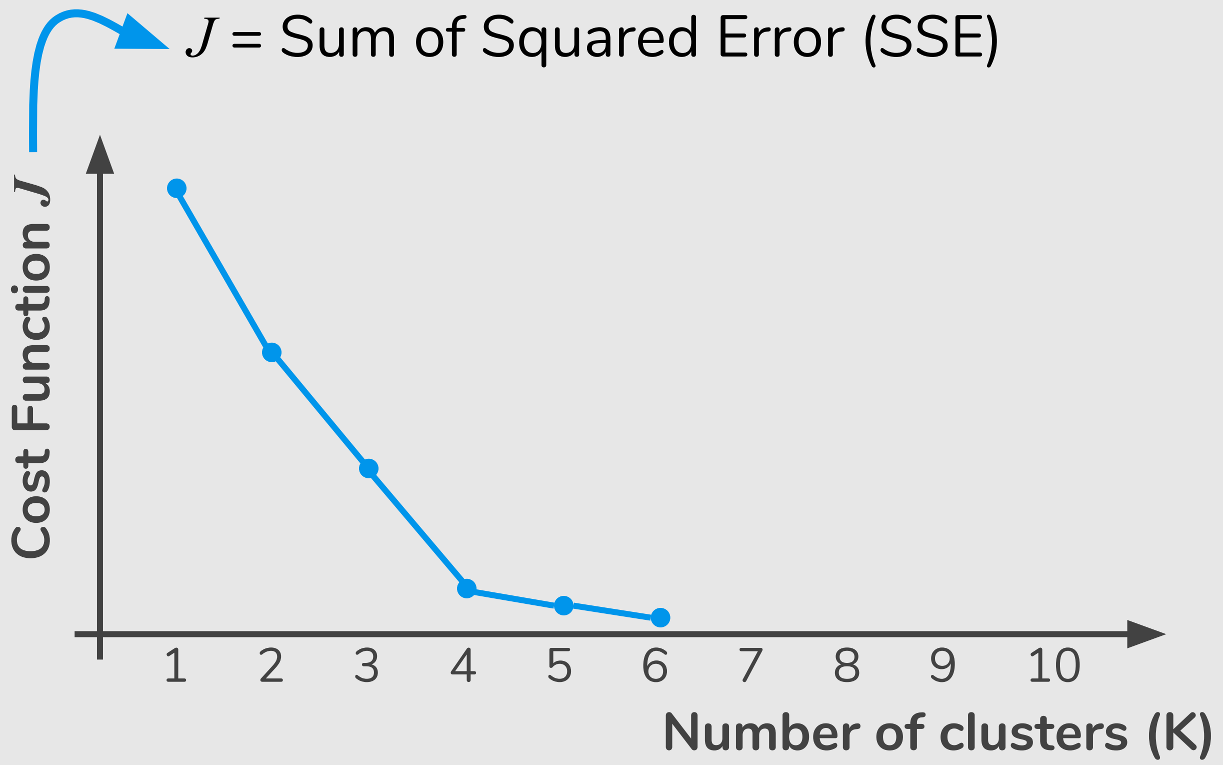
Elbow Method

J = Sum of Squared Error (SSE)



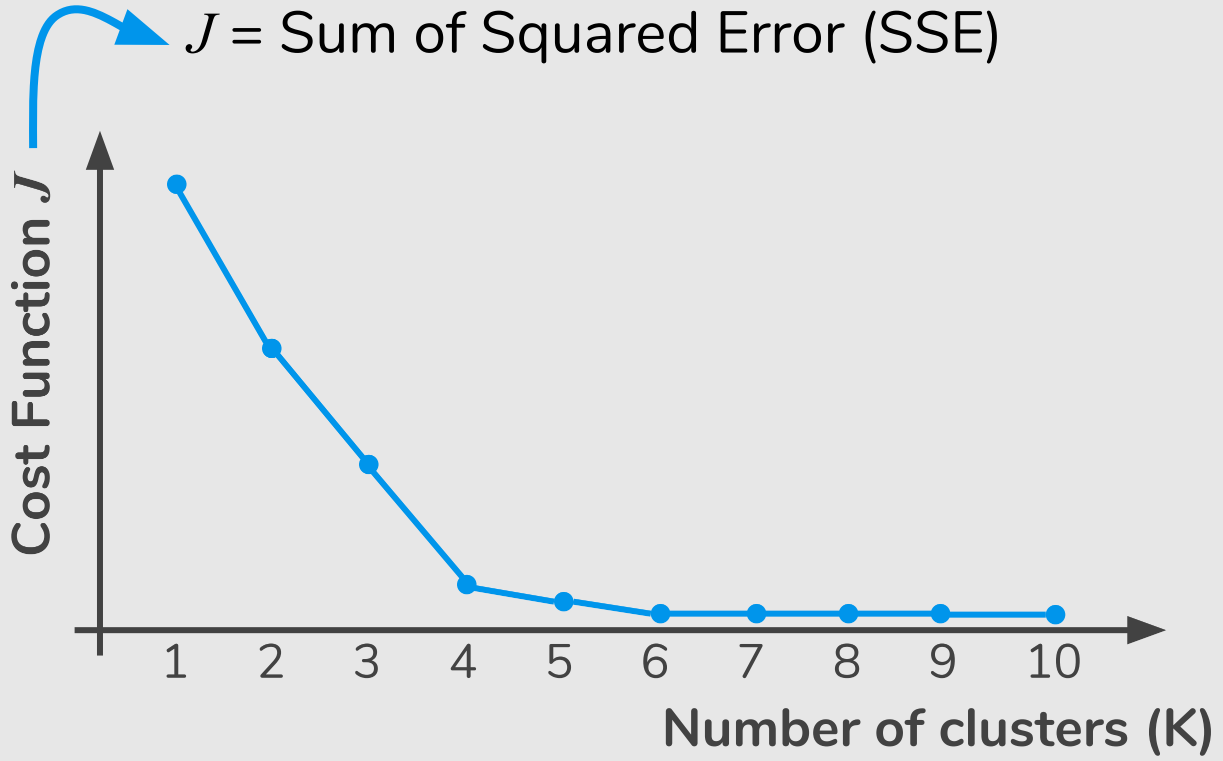
Elbow Method

J = Sum of Squared Error (SSE)

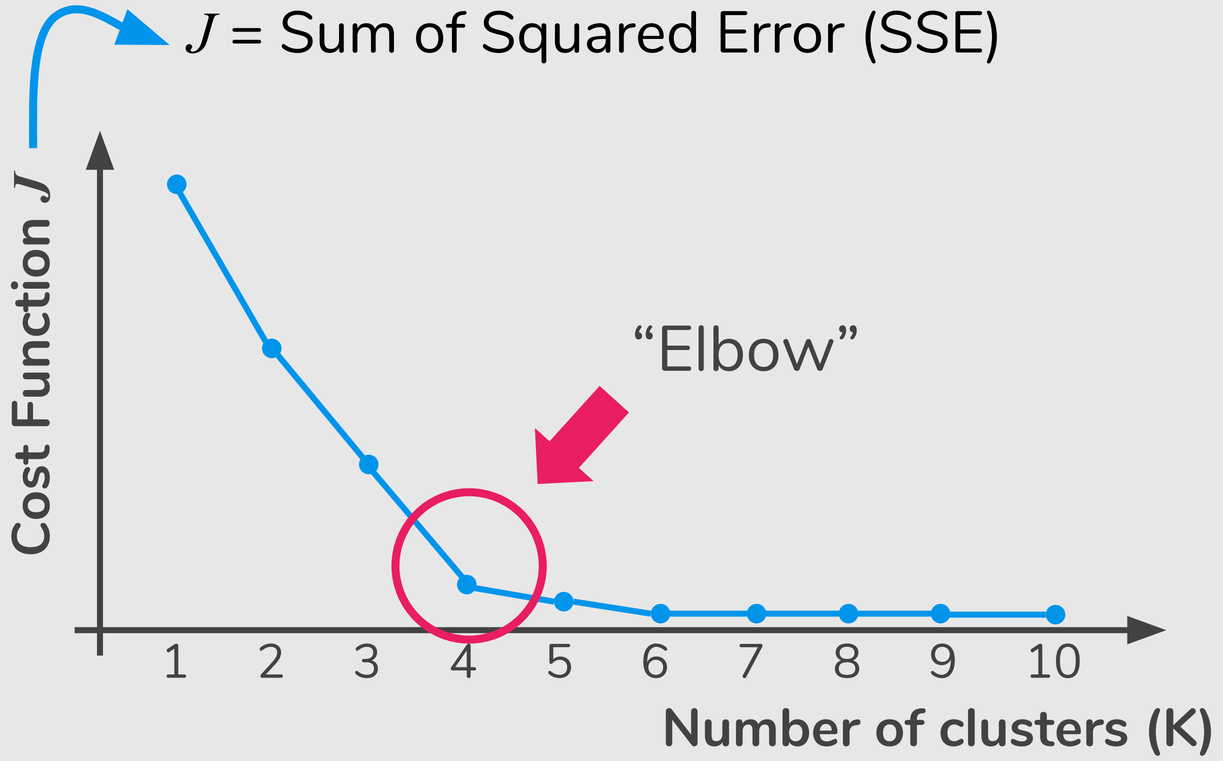


Elbow Method

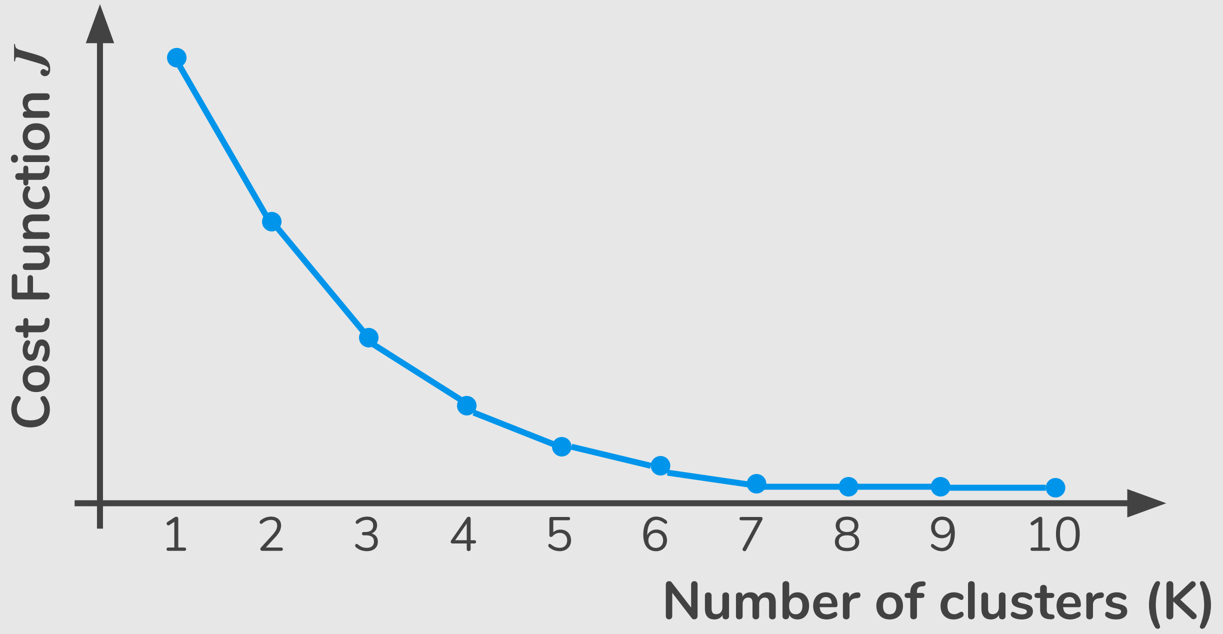
J = Sum of Squared Error (SSE)



Elbow Method

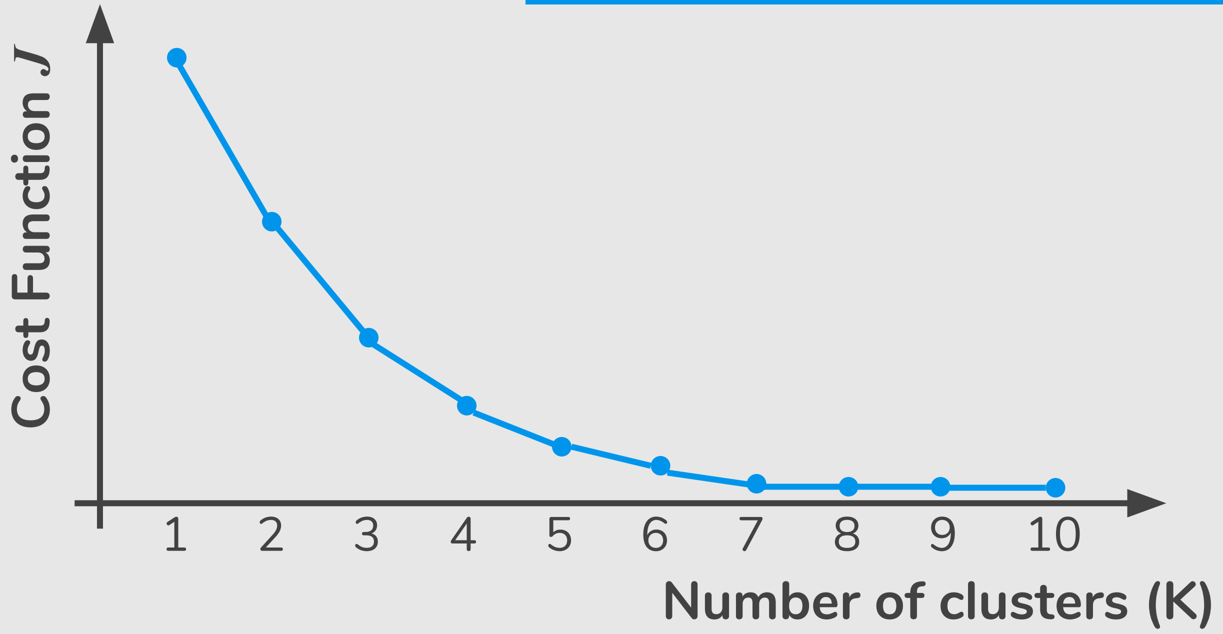


~~Elbow~~ Method

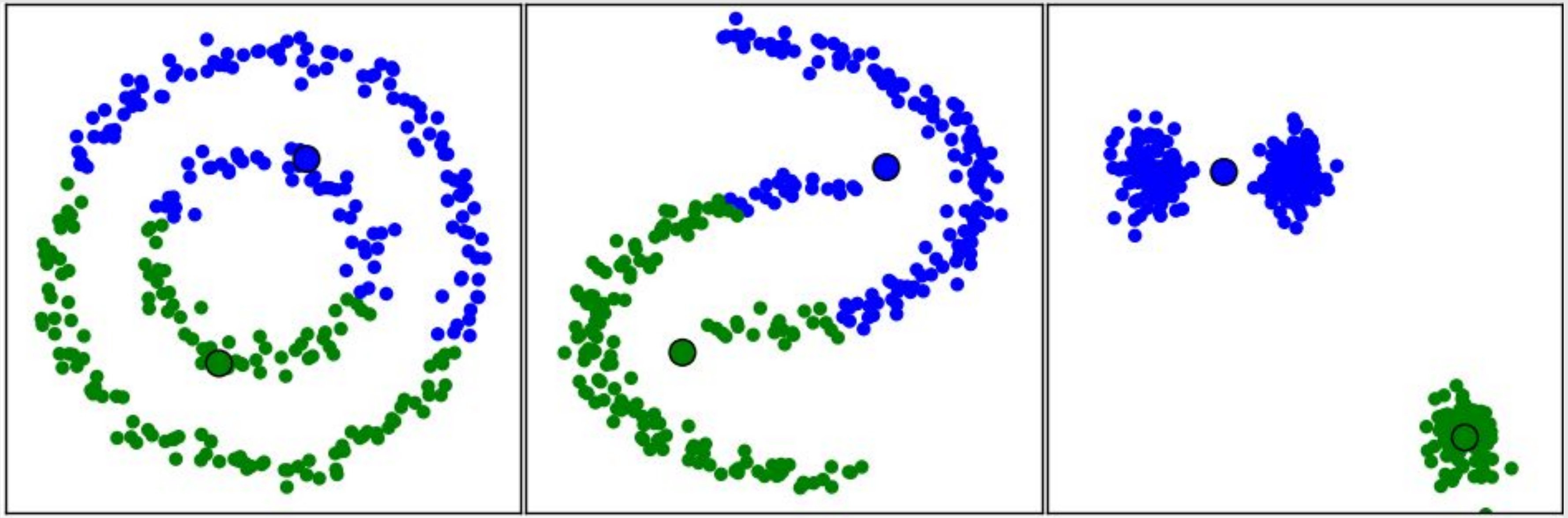


~~Elbow~~ Method

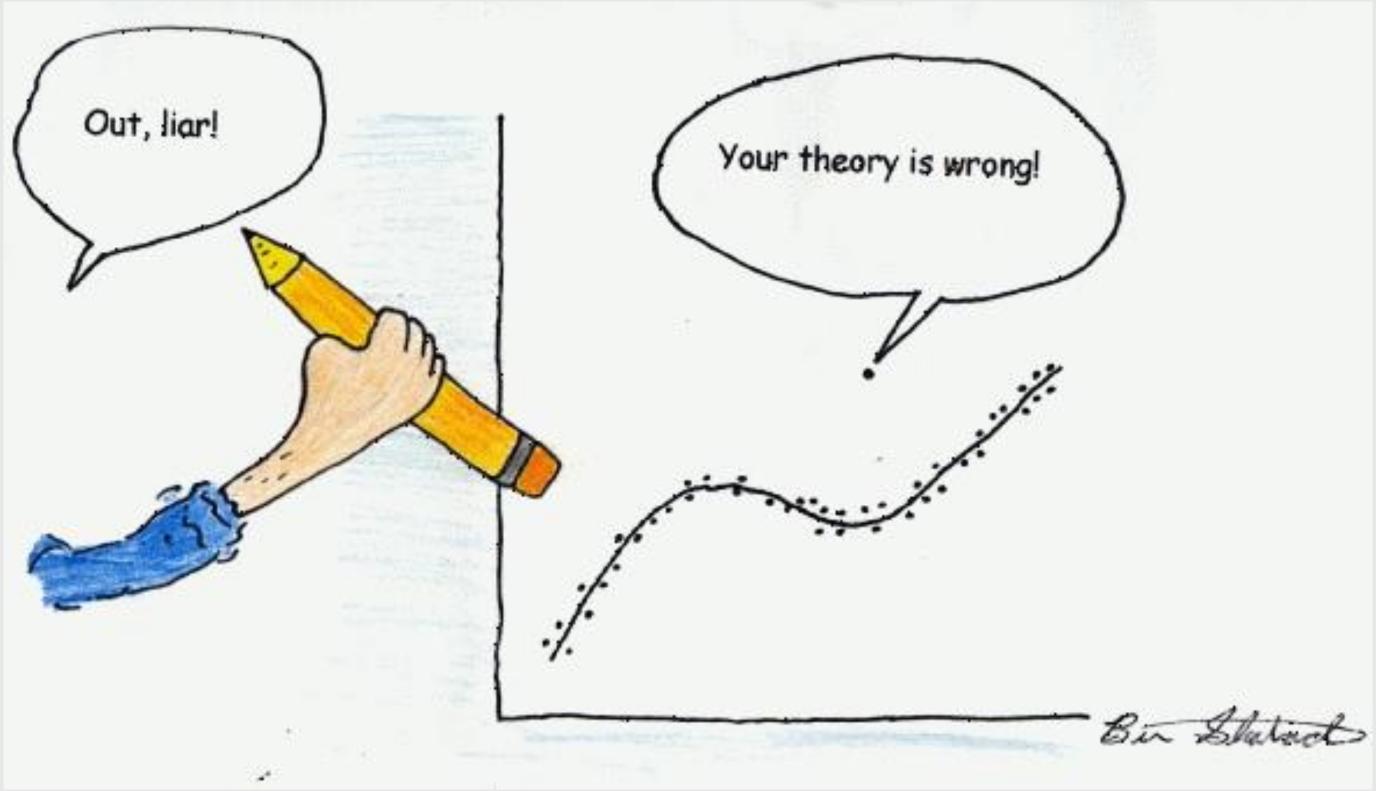
Q: You find that cost function J is much higher for k = 5 than for k =3. What can you conclude?



k-Means: Additional Issues



Outliers

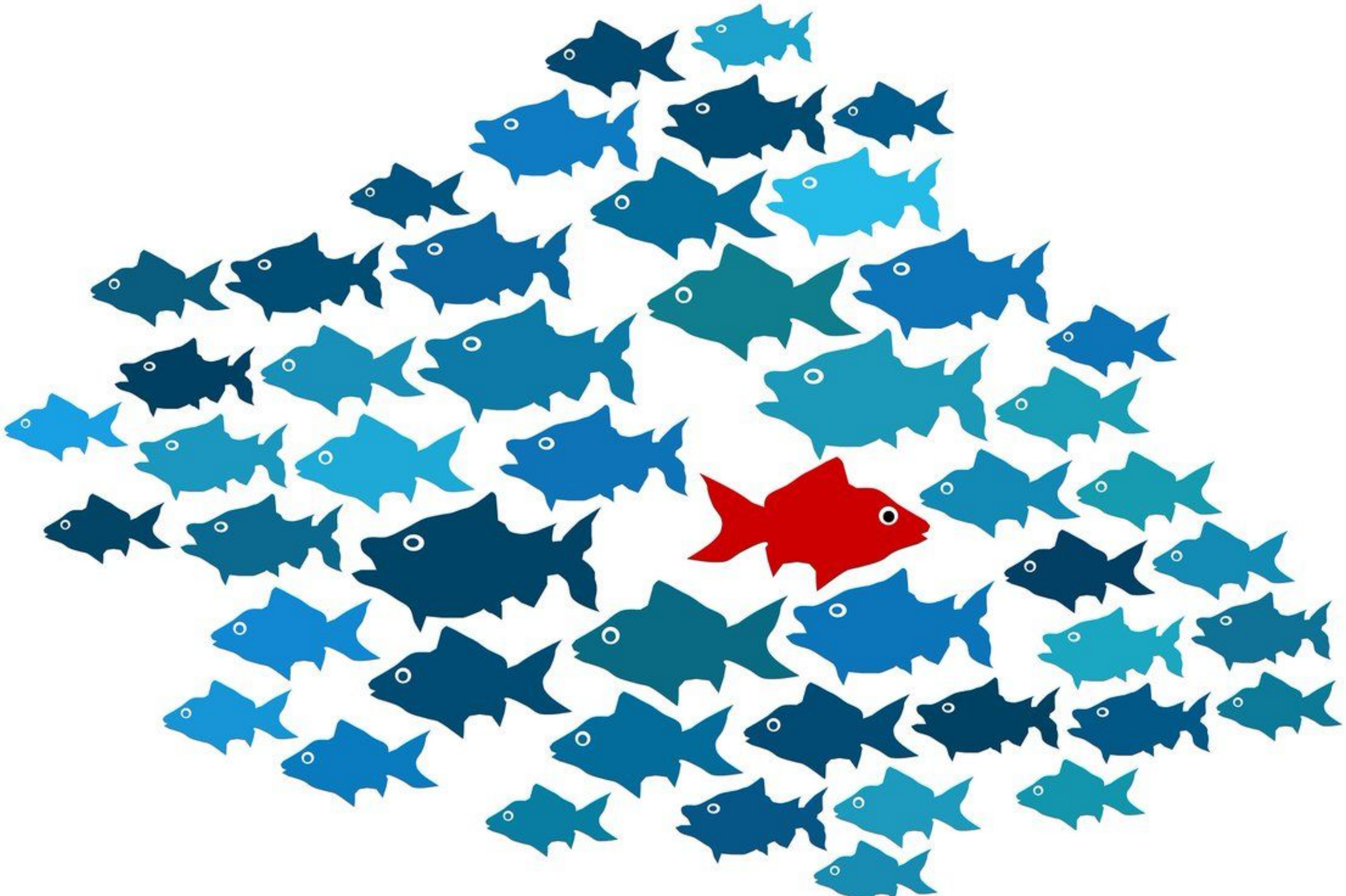


Outliers

- It is often useful to discover outliers and eliminate them before clustering.

Outliers

- It is often useful to discover outliers and eliminate them before clustering.
- Techniques for identifying outlier: “*Anomaly Detection*” *[chap. 9]*, *Introduction to Data Mining*, 2018.



Outliers

- It is often useful to discover outliers and eliminate them before clustering.
- Techniques for identifying outlier: “*Anomaly Detection*” *[chap. 9]*, *Introduction to Data Mining*, 2018.
- Also, we often want to eliminate small clusters because they frequently represent groups of outliers.

Reducing the SSE with Postprocessing

- **Split a cluster**: the cluster with the largest SSE is usually chosen.

Reducing the SSE with Postprocessing

- **Split a cluster**: the cluster with the largest SSE is usually chosen.
- **Introduce a new cluster centroid**: often the point that is farthest from any cluster center is chosen.

Reducing the SSE with Postprocessing

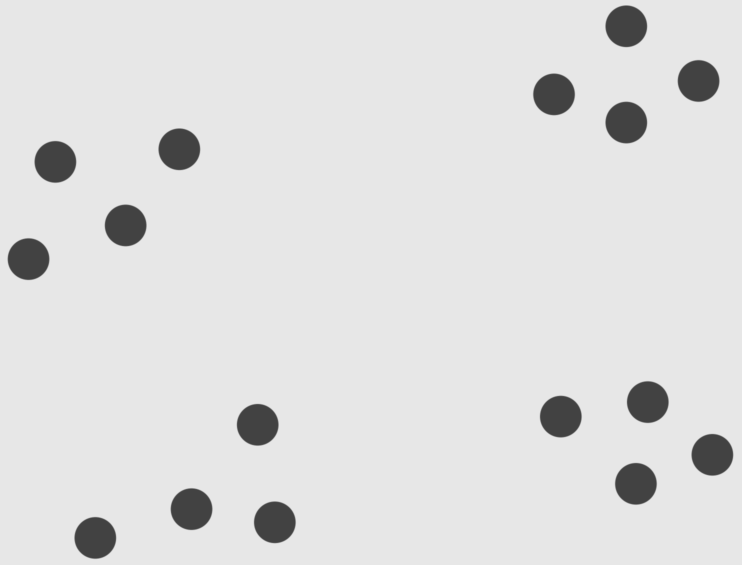
- **Split a cluster**: the cluster with the largest SSE is usually chosen.
- **Introduce a new cluster centroid**: often the point that is farthest from any cluster center is chosen.
- **Merge two clusters**: The clusters with the closest centroids are typically chosen.

k-Means Variations

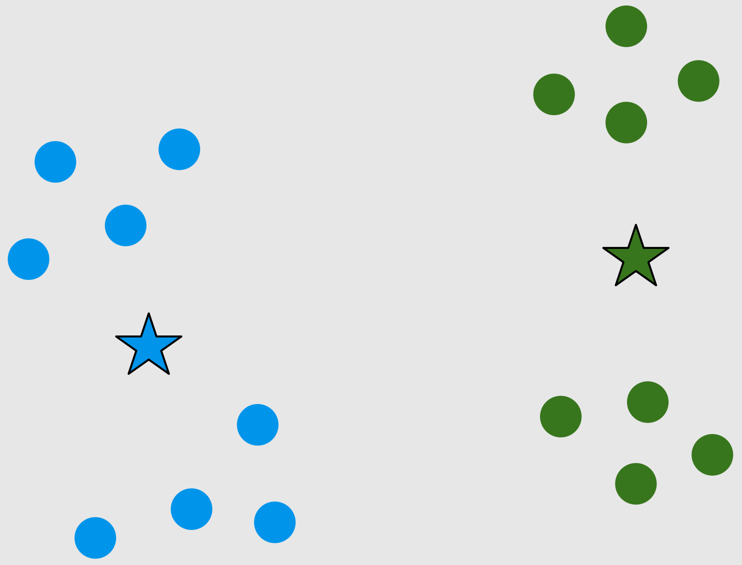
Bisecting k-Means

- A straightforward extension of the basic k-means.
- To obtain k clusters:
  - Split the set of all points into two clusters,
  - Select one of these clusters to split,
  - Repeat until k clusters have been produced.

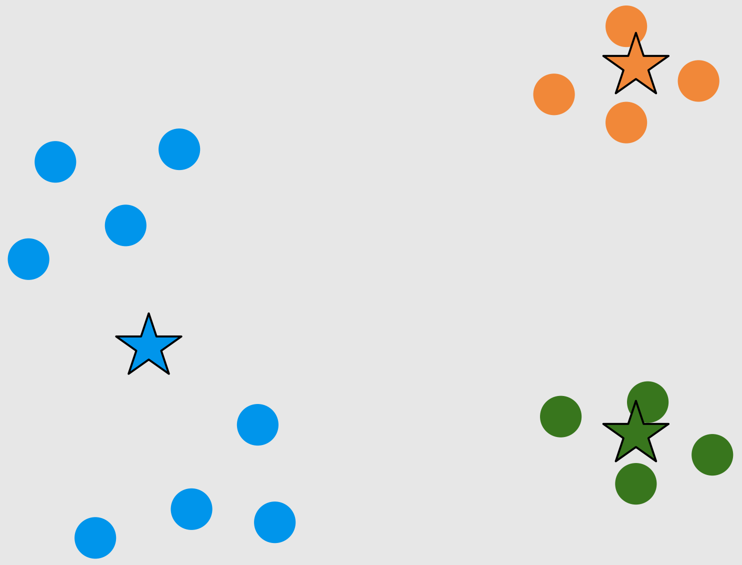
Bisecting k-Means



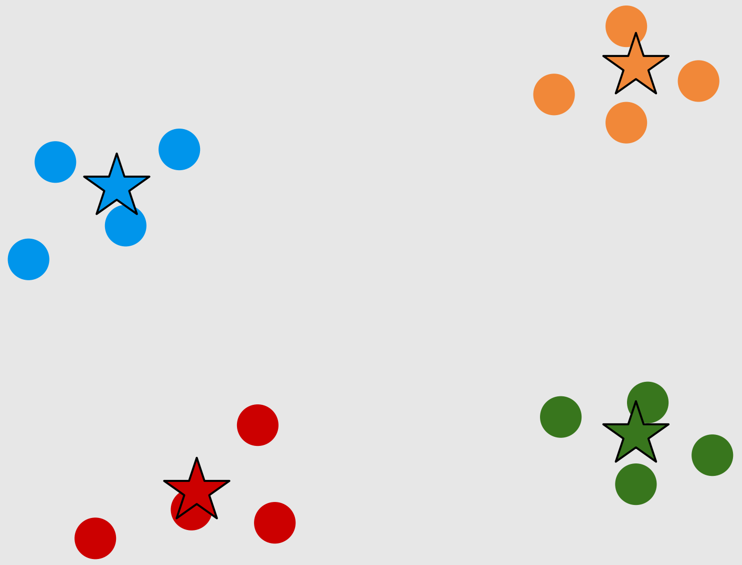
Bisecting k-Means



Bisecting k-Means



Bisecting k-Means



Mini-batch k-Means

- Uses mini-batches to reduce the computation time, while still attempting to optimize the same objective function.

- Converges faster than k-Means, but the quality of the results is reduced.

k-Medians Clustering

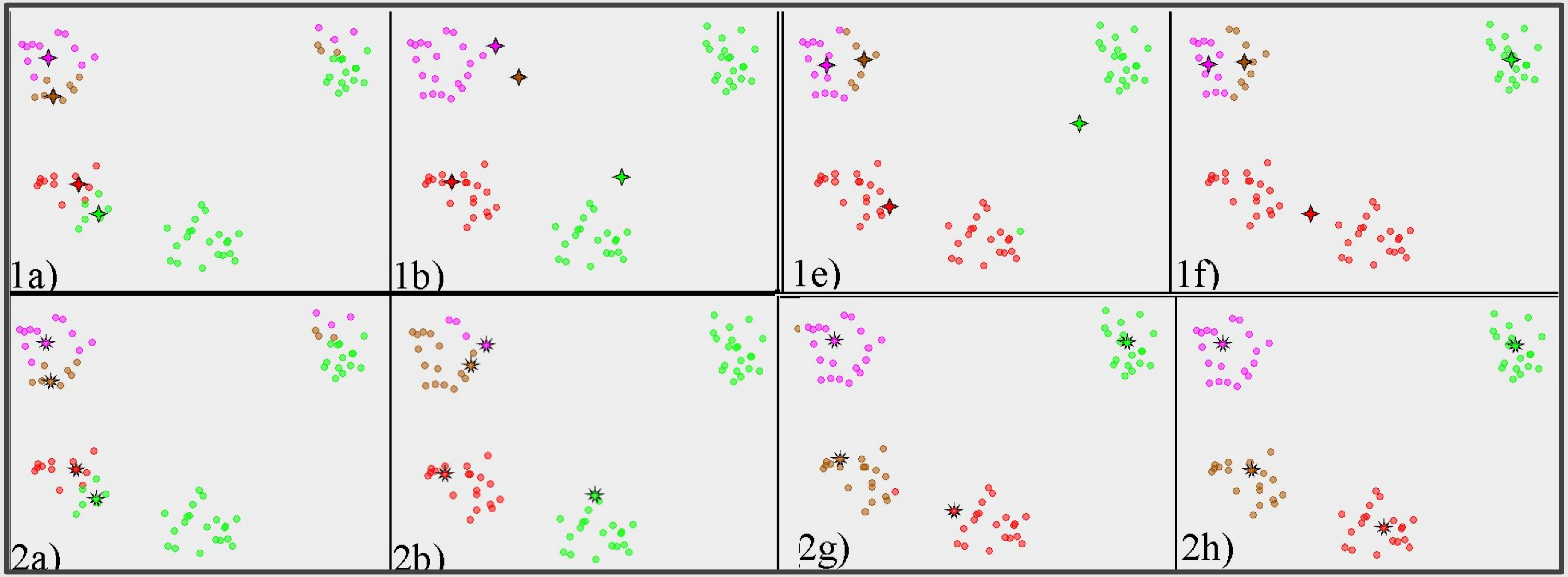
- Instead of calculating the mean for each cluster to determine its centroid, one instead **calculates the median**.

- Minimizing error over all clusters with respect to the **1-norm distance metric**, as opposed to the square of the 2-norm distance metric (which k-Means does).

k-Medoids Clustering

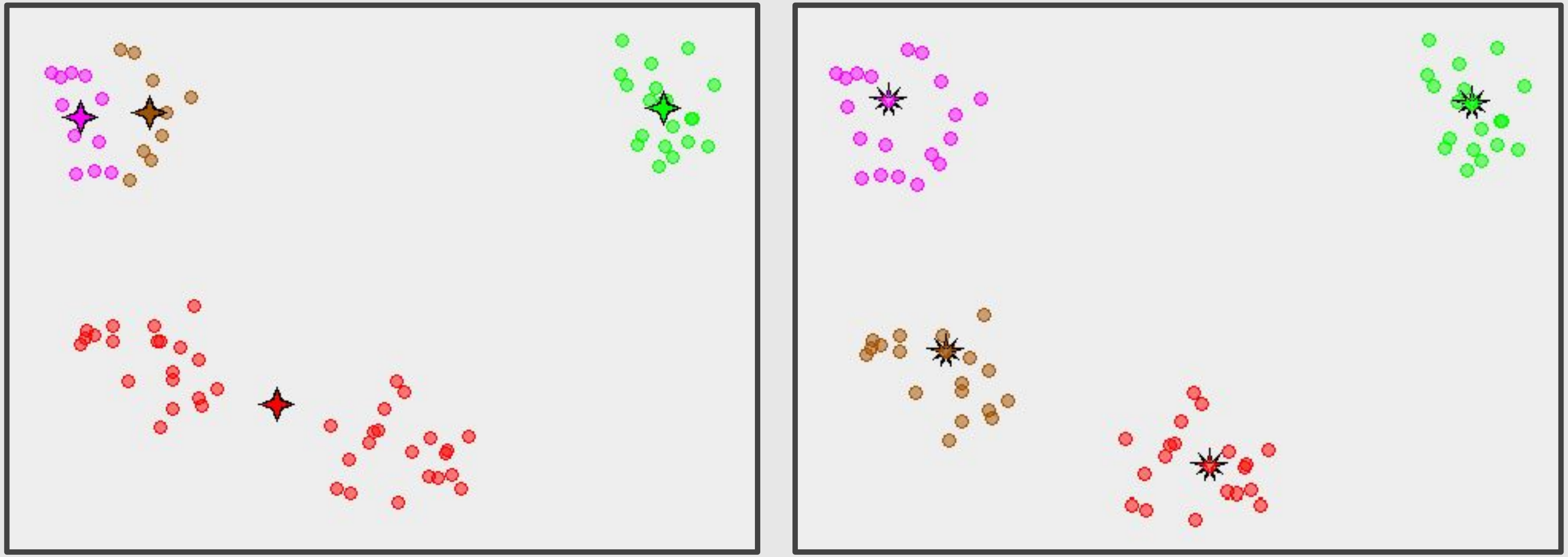
- Instead of calculating the mean for each cluster to determine its centroid, one instead **calculates the medoid**.
- Minimizing error over all clusters with respect to the **1-norm distance metric**.
- In contrast to the k-Means, k-Medoids **chooses data points as centroids**.

k-Means (top) vs k-Medoids (bottom)



Credit: https://commons.wikimedia.org/wiki/File:K-means_versus_k-medoids.png

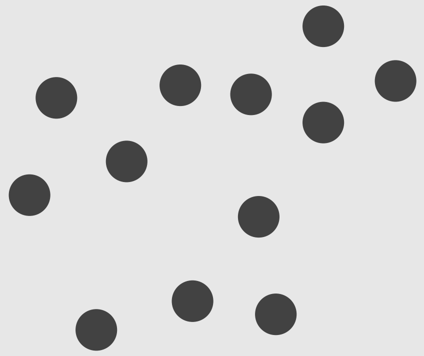
k-Means (left) vs k-Medoids (right)

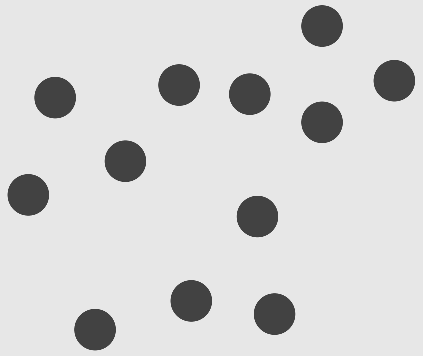


Credit: https://commons.wikimedia.org/wiki/File:K-means_versus_k-medoids.png

Fuzzy Clustering (Soft Clustering)

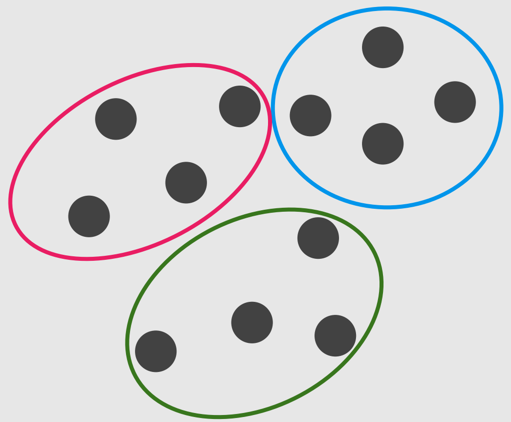
- Each data point can belong to more than one cluster.

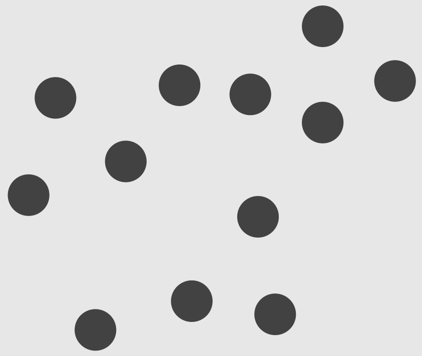

Hard clustering


Soft clustering

Fuzzy Clustering (Soft Clustering)

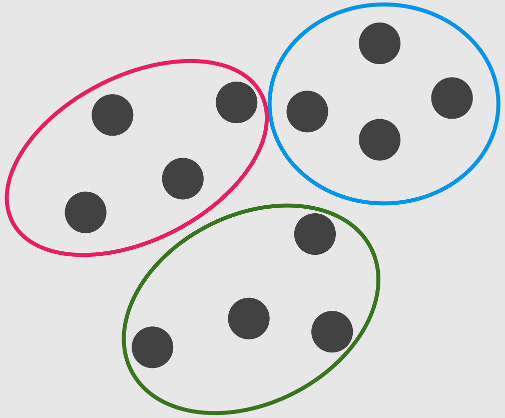
- Each data point can belong to more than one cluster.

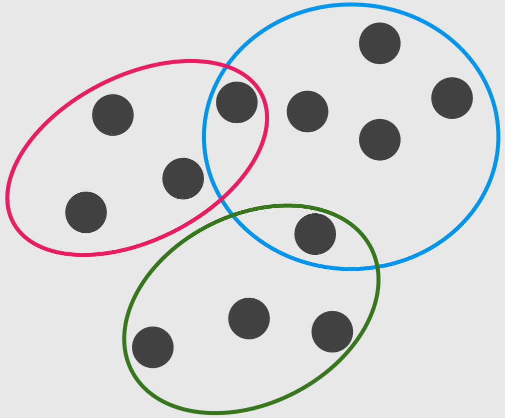

Hard clustering


Soft clustering

Fuzzy Clustering (Soft Clustering)

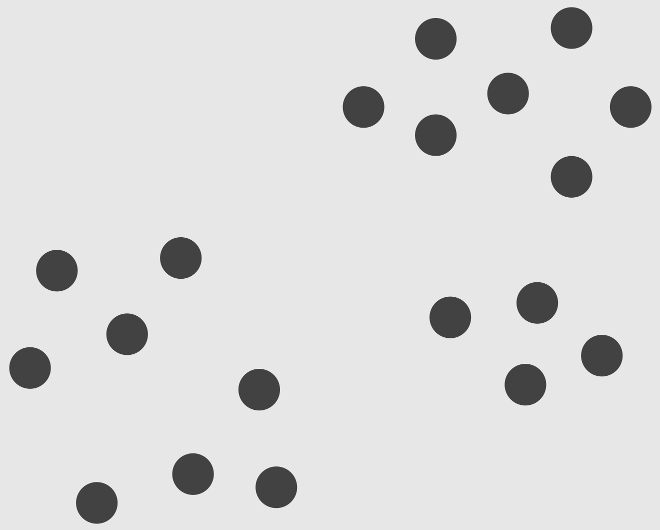
- Each data point can belong to more than one cluster.


Hard clustering

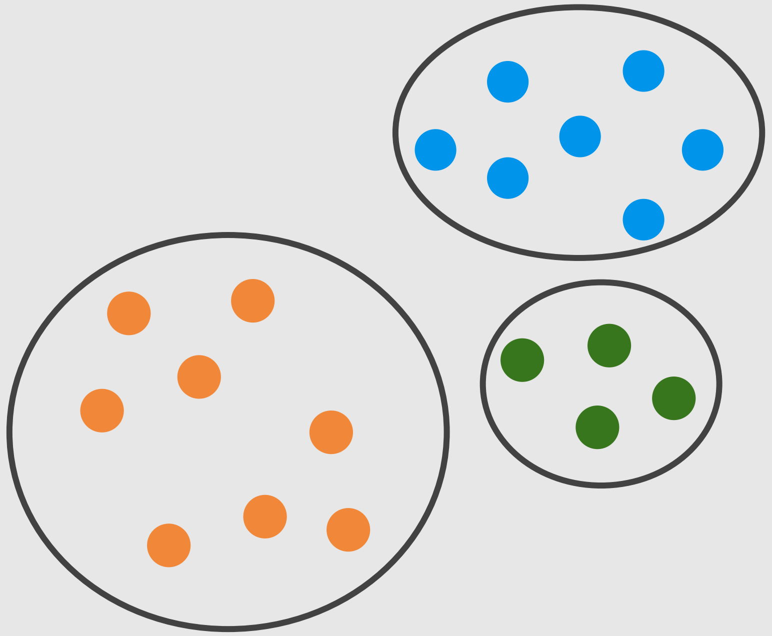

Soft clustering

Hierarchical Clustering

Hierarchical versus Partitional

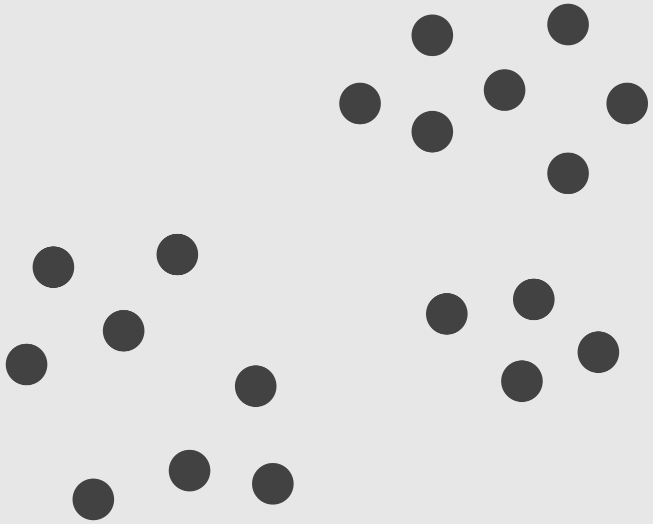


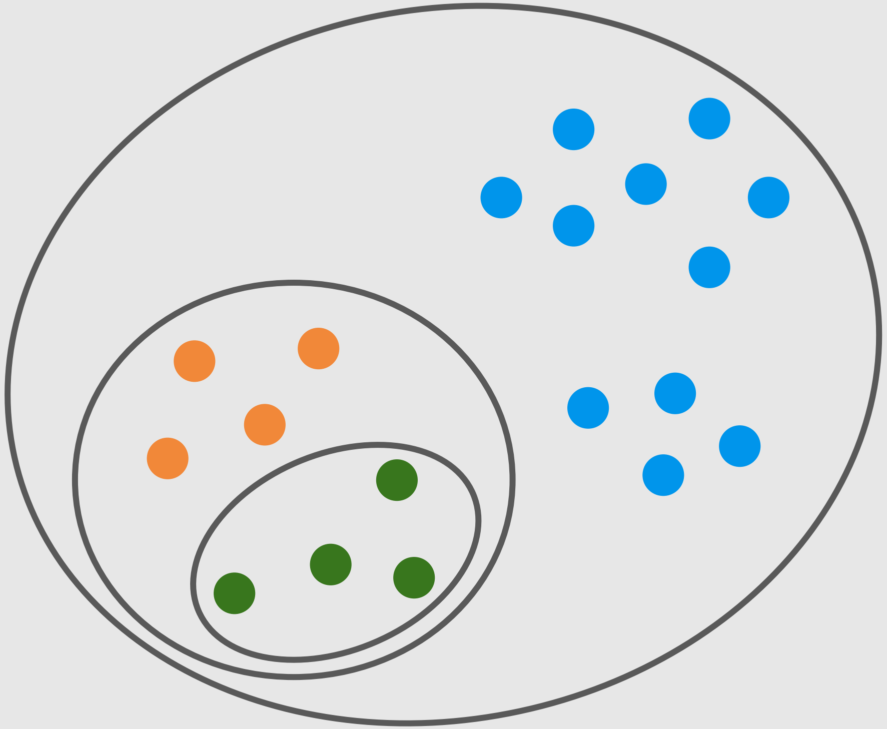
Original data



Partitional clustering

Hierarchical versus Partitional


Original data

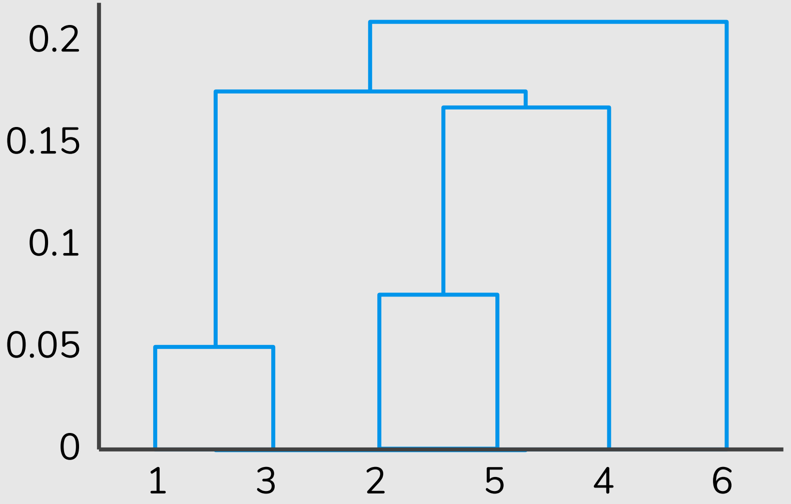

Hierarchical clustering

Hierarchical Clustering

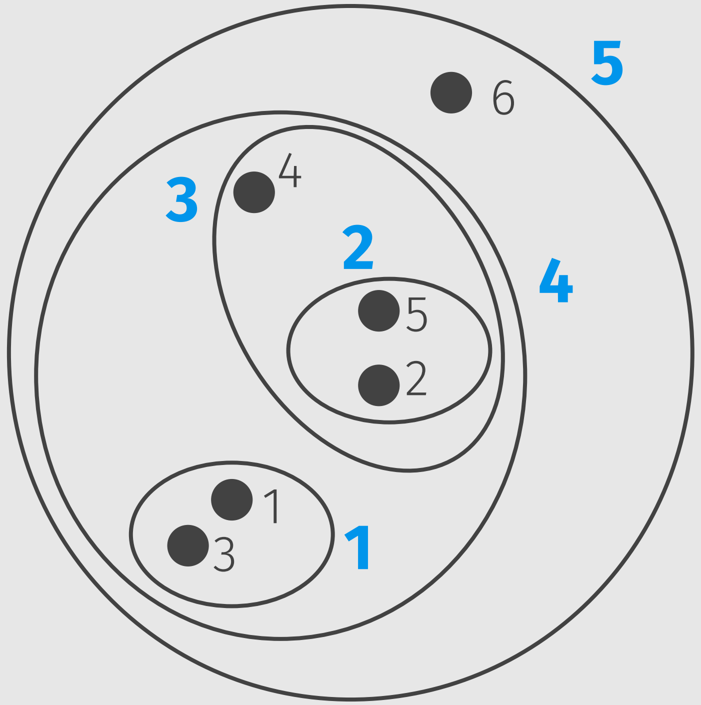
- **Agglomerative** ("bottom up"): each observation starts in its own cluster, and pairs of clusters are merged as one moves up the hierarchy.

- **Divisive** ("top down"): all observations start in one cluster, and splits are performed recursively as one moves down the hierarchy.

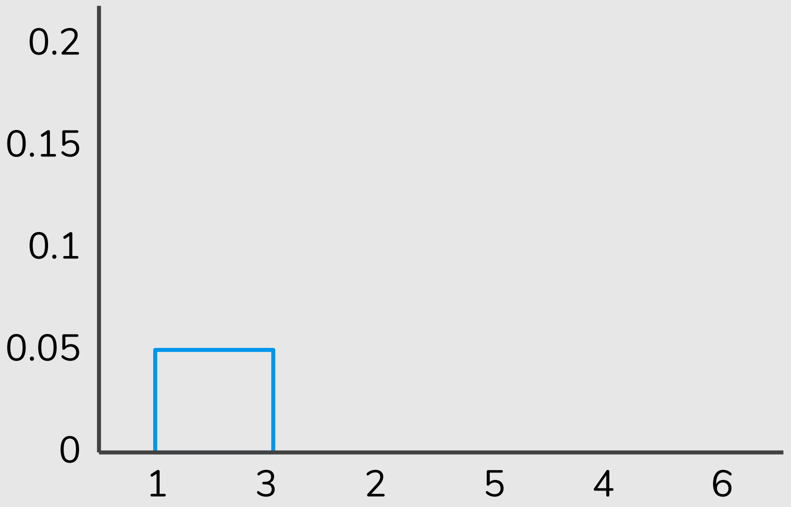
Agglomerative Hierarchical Clustering



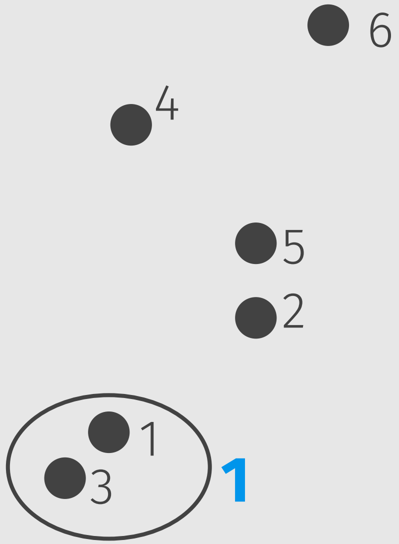
Dendrogram



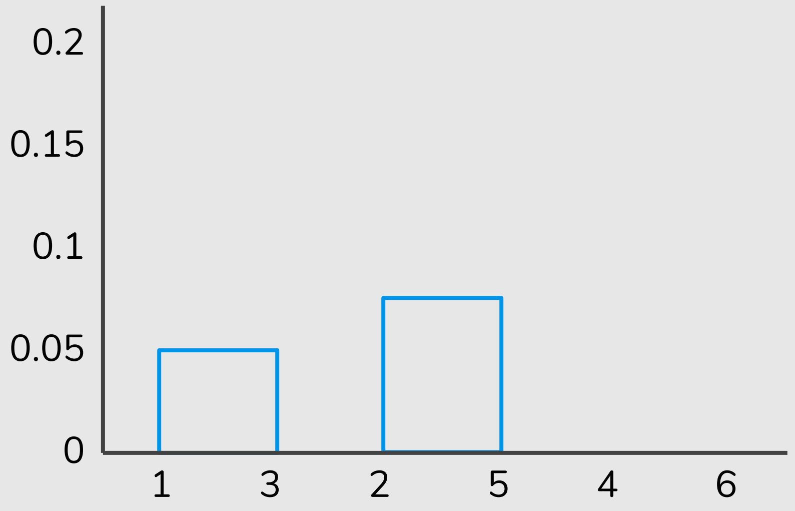
Agglomerative Hierarchical Clustering



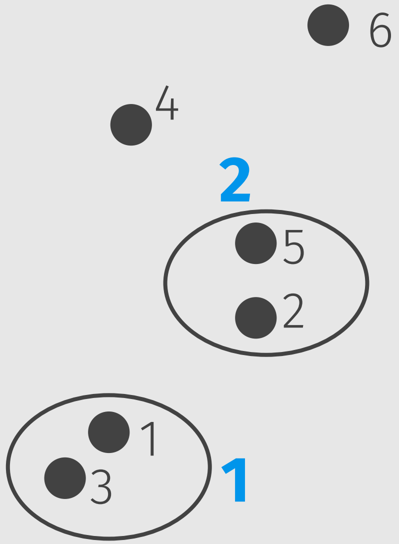
Dendrogram



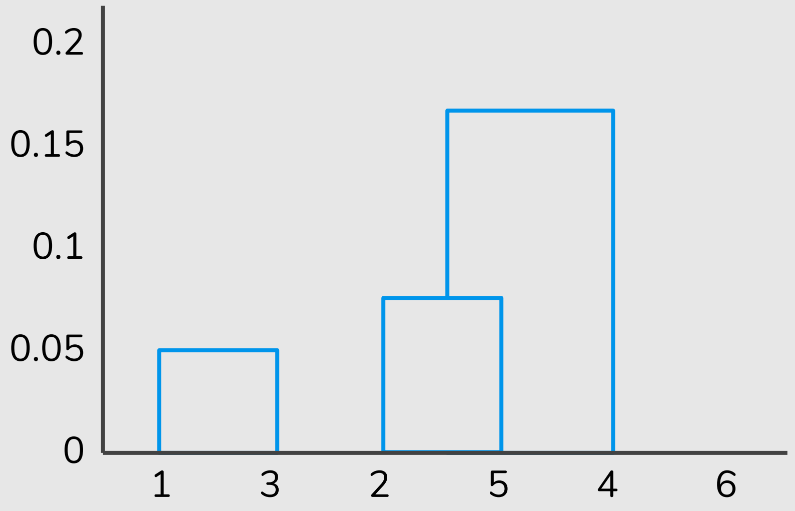
Agglomerative Hierarchical Clustering



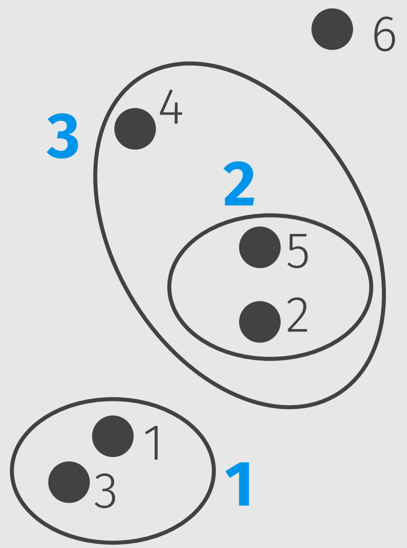
Dendrogram



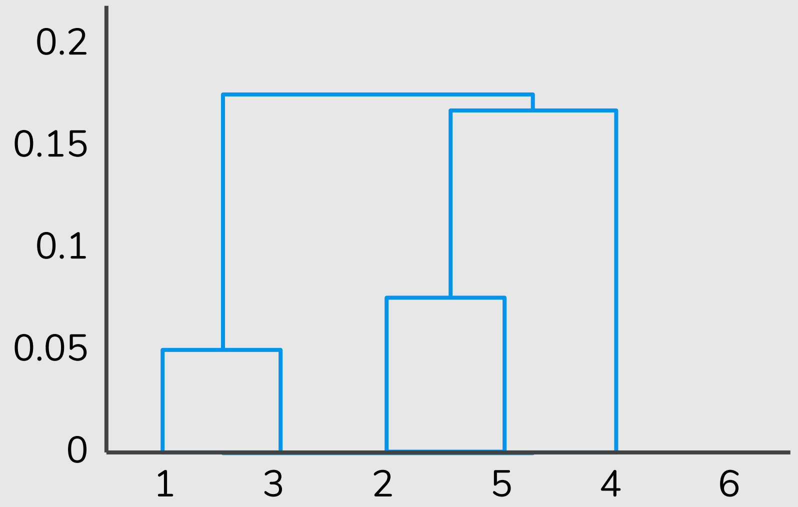
Agglomerative Hierarchical Clustering



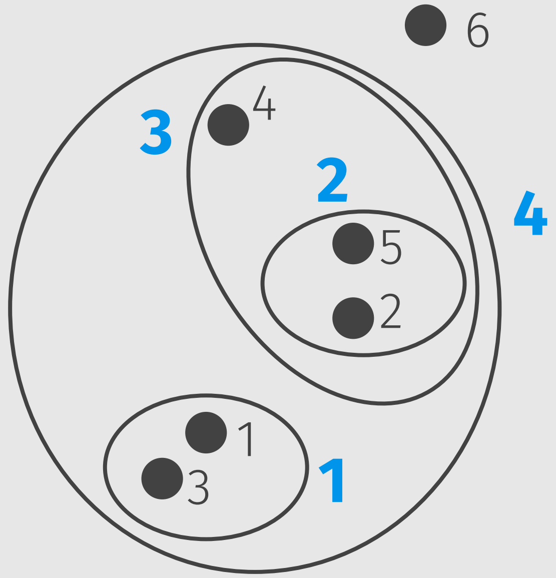
Dendrogram



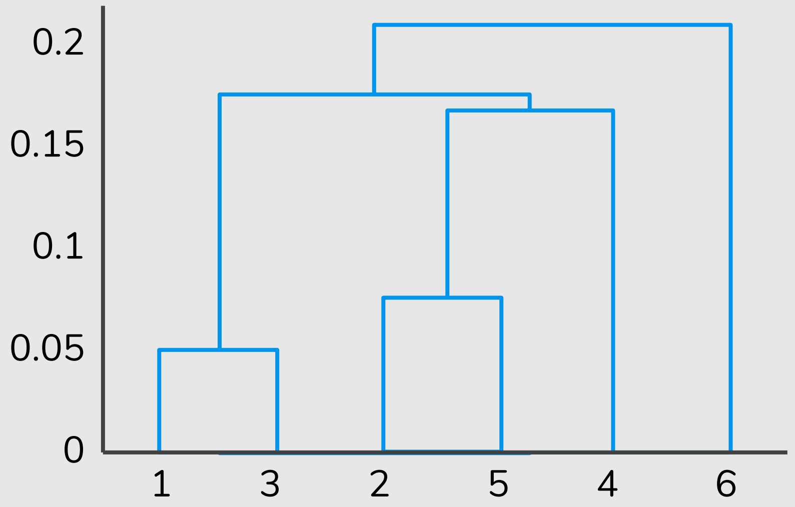
Agglomerative Hierarchical Clustering



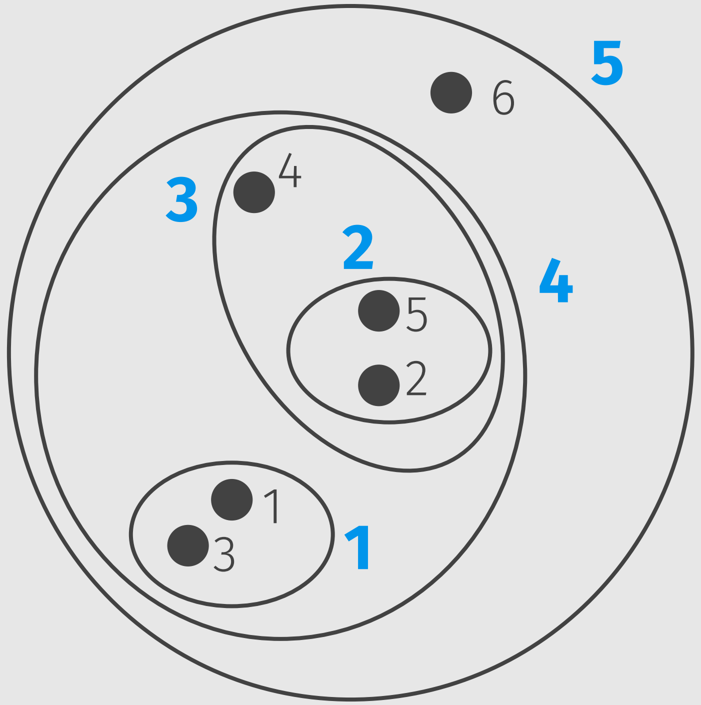
Dendrogram



Agglomerative Hierarchical Clustering



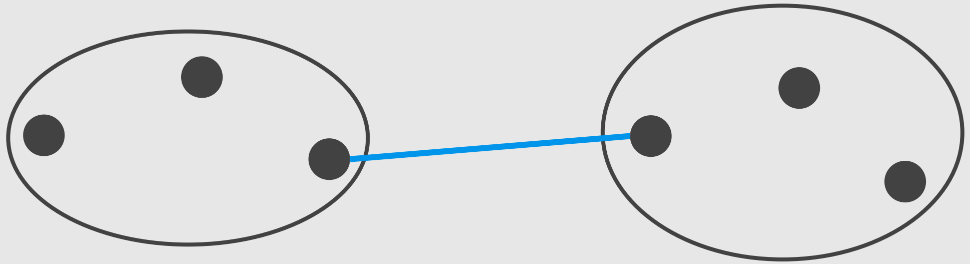
Dendrogram



Agglomerative Hierarchical Clustering

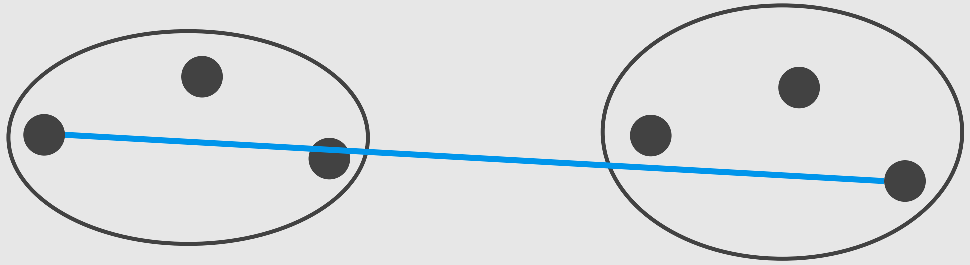
```
1: compute the **proximity matrix**, if necessary.
2: **repeat**
3:     merge the closest two clusters.
4:     update the proximity matrix to reflect the proximity
       between the new cluster and the original clusters.
5: **until** only one cluster remains.
```

Defining Proximity between Clusters



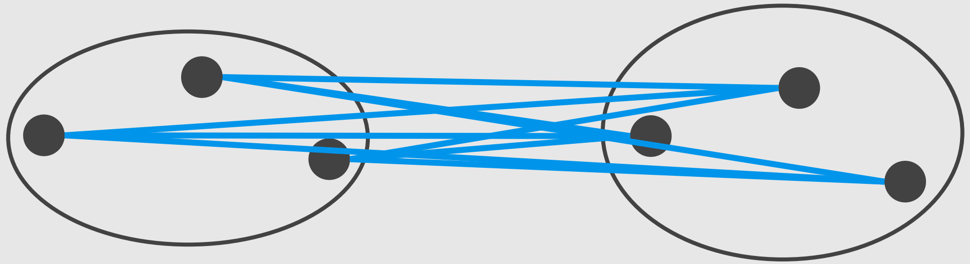
Single link or **MIN**: defines cluster proximity as the **proximity** between the closest two points that are in different clusters.

Defining Proximity between Clusters



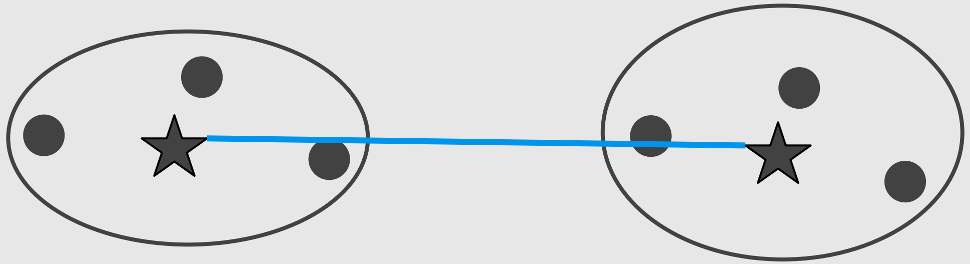
Complete link or **MAX**: takes the proximity between the **farthest** two points in different clusters to be the cluster proximity.

Defining Proximity between Clusters



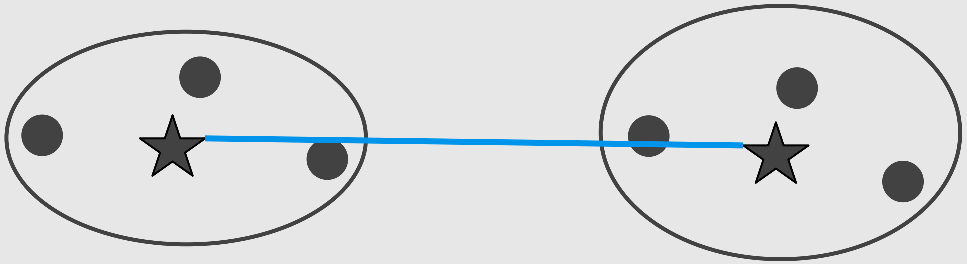
Average: defines cluster proximity to be the **average pairwise** proximities of all pairs of points from different clusters.

Defining Proximity between Clusters



Centroids: the cluster proximity is commonly defined as the proximity between cluster centroids.

Defining Proximity between Clusters

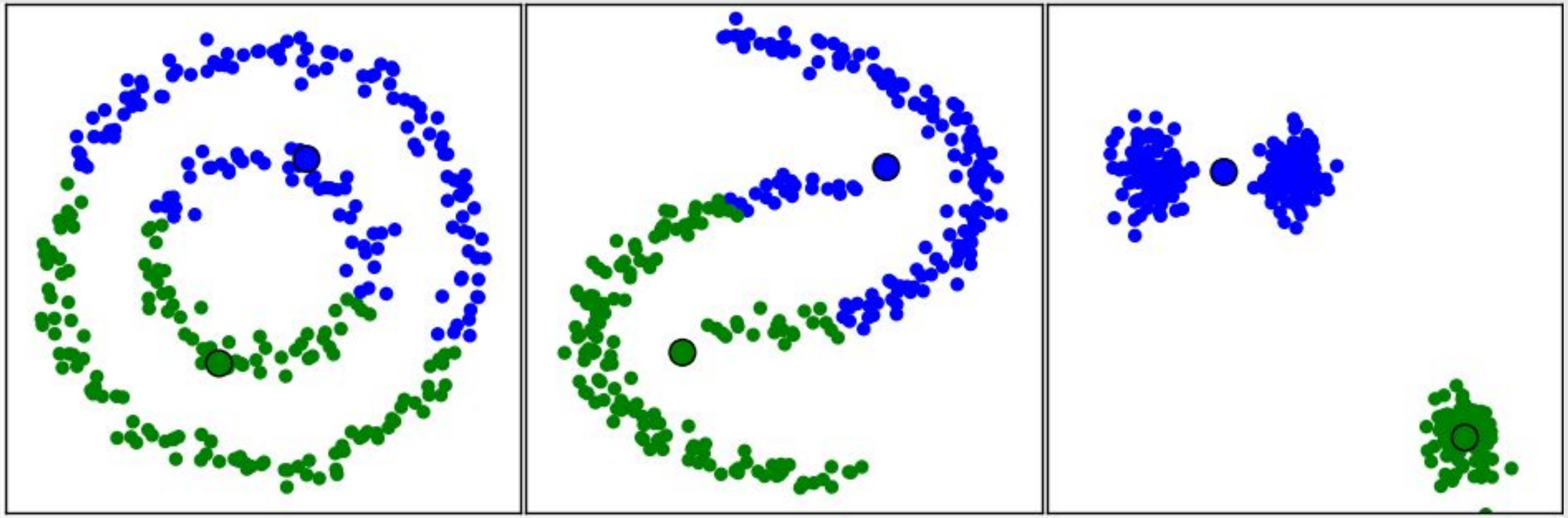


Ward's: measures the proximity between two clusters in terms of the increase in the SSE that results from merging the two cluster.

Agglomerative Hierarchical Clustering

```
1: compute the proximity matrix, if necessary.
2: repeat
3:       merge the closest two clusters.
4:       update the proximity matrix to reflect the proximity
         between the new cluster and the original clusters.
5: until only one cluster remains.
```

DBSCAN

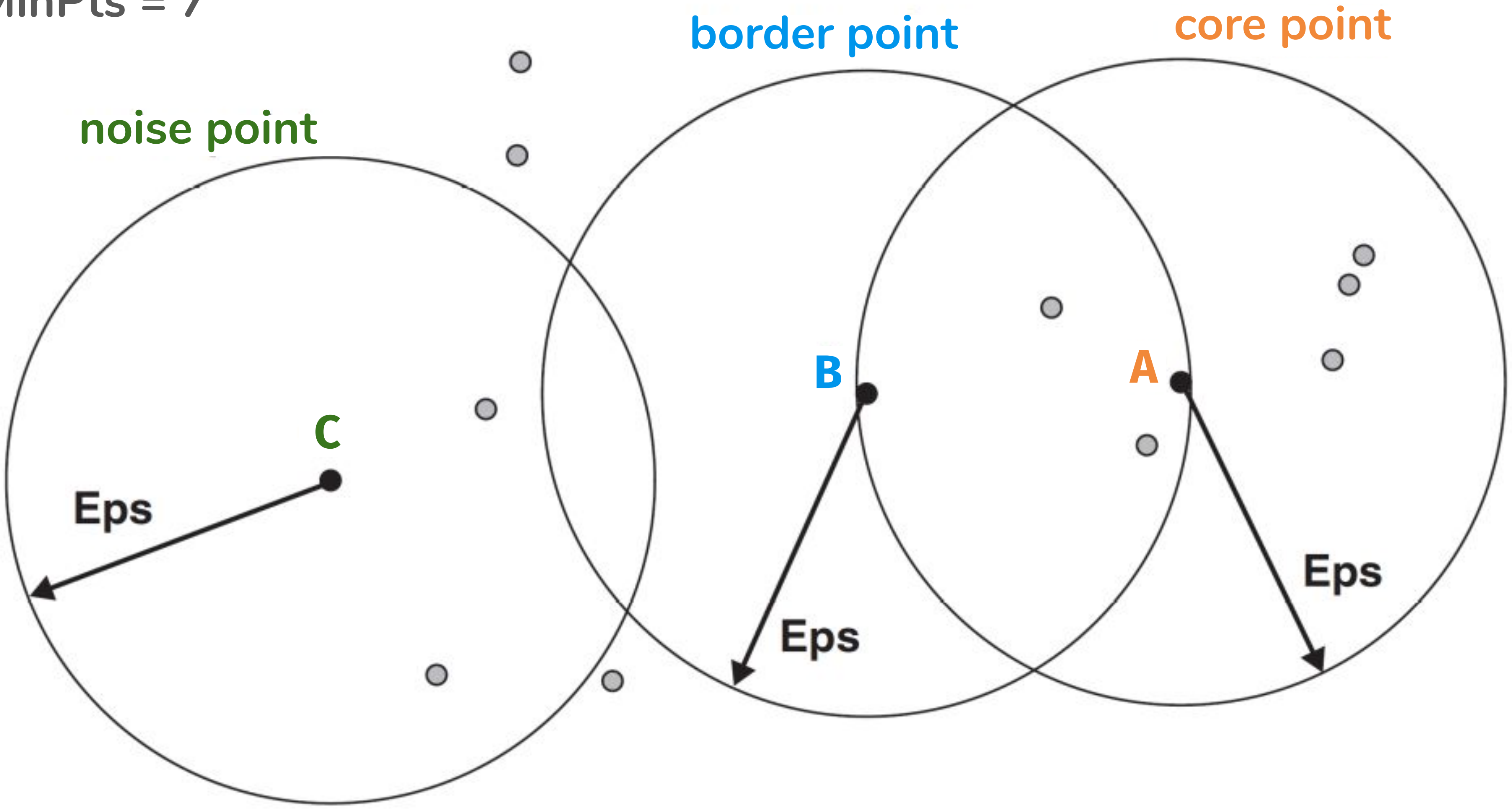


DBSCAN Clustering

- **D**ensity-**B**ased **S**patial **C**lustering of **A**pplications with **N**oise

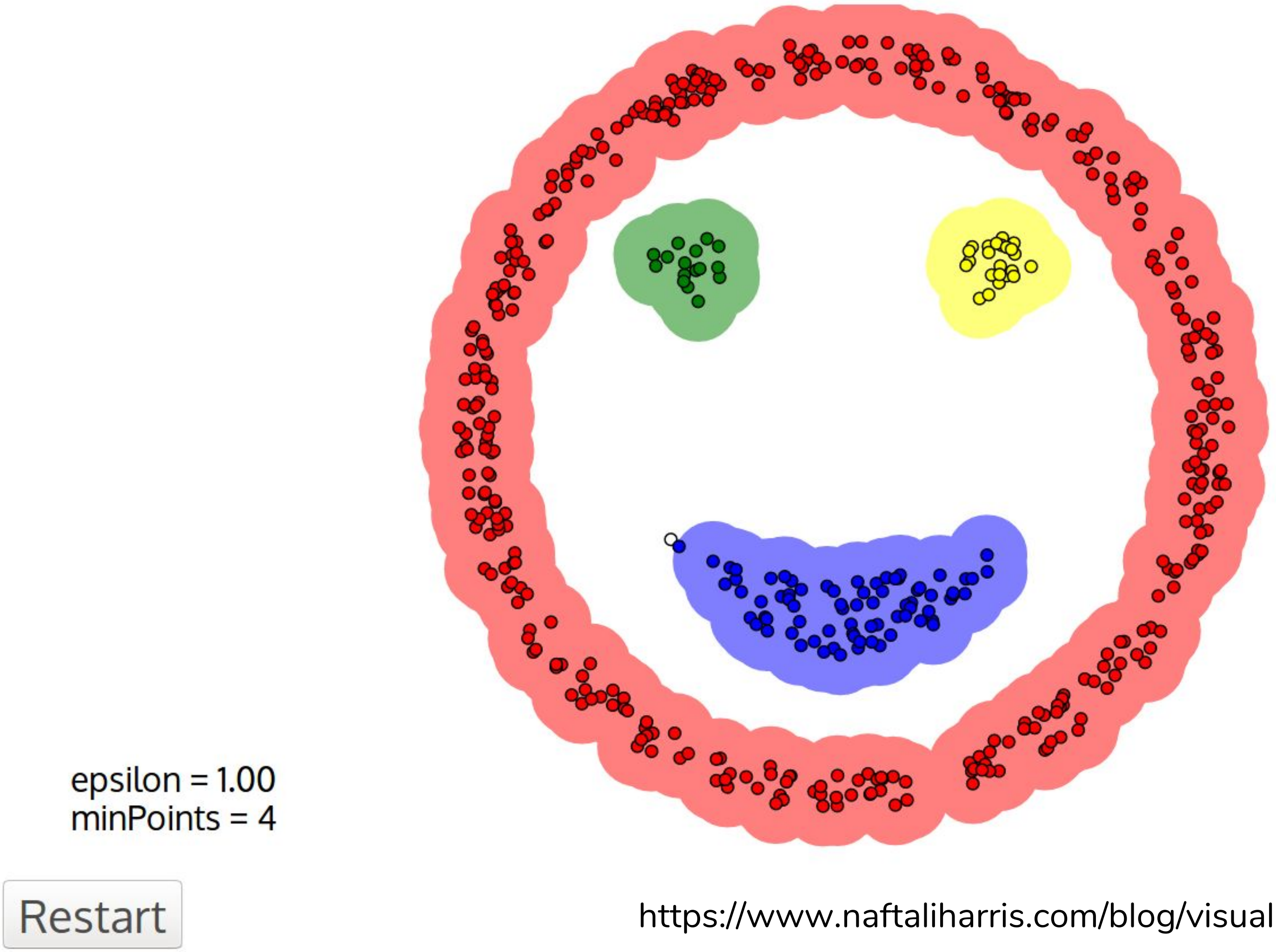
- Given a set of points in some space, **it groups together points that are closely packed together** (points with many nearby neighbors), marking as outliers points that lie alone in low-density regions.

MinPts = 7



DBSCAN Clustering

- **Core points**: A point is a core point if there are at least *MinPts* within a distance of *Eps*, where *MinPts* and *Eps* are user-specified parameters.
- **Border points**: A border point is not a core point, but falls within the neighborhood of a core point.
- **Noise points**: A noise point is any point that is neither a core point nor a border point.


epsilon = 1.00
minPoints = 4

https://www.naftaliharris.com/blog/visualizing-dbscan-clustering

Clustering Performance Evaluation

Clustering Evaluation

- Evaluating the performance of a clustering algorithm **is not as trivial** as counting the number of errors or the precision and recall of a supervised classification algorithm.

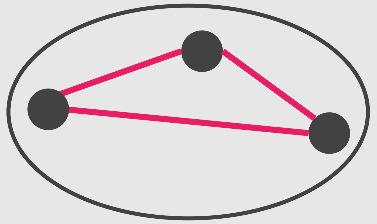
Clustering Evaluation

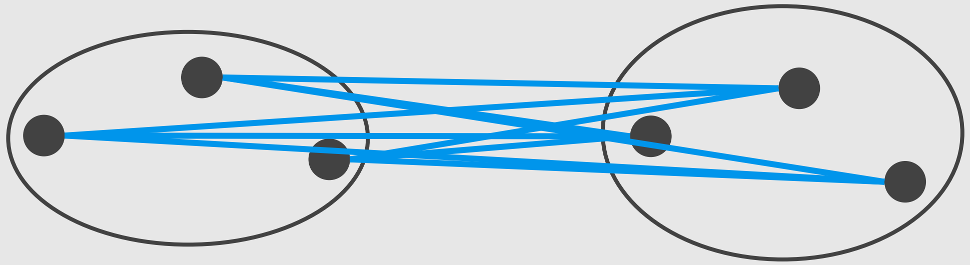
- Evaluating the performance of a clustering algorithm **is not as trivial** as counting the number of errors or the precision and recall of a supervised classification algorithm.

- Adjusted Rand index
- Mutual Information based scores
- Homogeneity, completeness and V-measure
- **Silhouette Coefficient**

Silhouette Coefficient

- The silhouette value is a measure of how similar a sample is to its own cluster (**cohesion**) compared to other clusters (**separation**).


Cohesion


Separation

Silhouette Coefficient

- The silhouette value is a measure of how similar a sample is to its own cluster (**cohesion**) compared to other clusters (**separation**).

- The silhouette ranges from −1 to +1.
    - High value = the clustering configuration is appropriate.
    - Low value = the clustering configuration may have too many or too few clusters.

Silhouette Coefficient

- The Silhouette Coefficient is defined **for each sample** and is composed of two scores:
  - a: The mean distance between a sample and all other points **in the same cluster**.
  - b: The mean distance between a sample and all other points **in the next nearest cluster**.

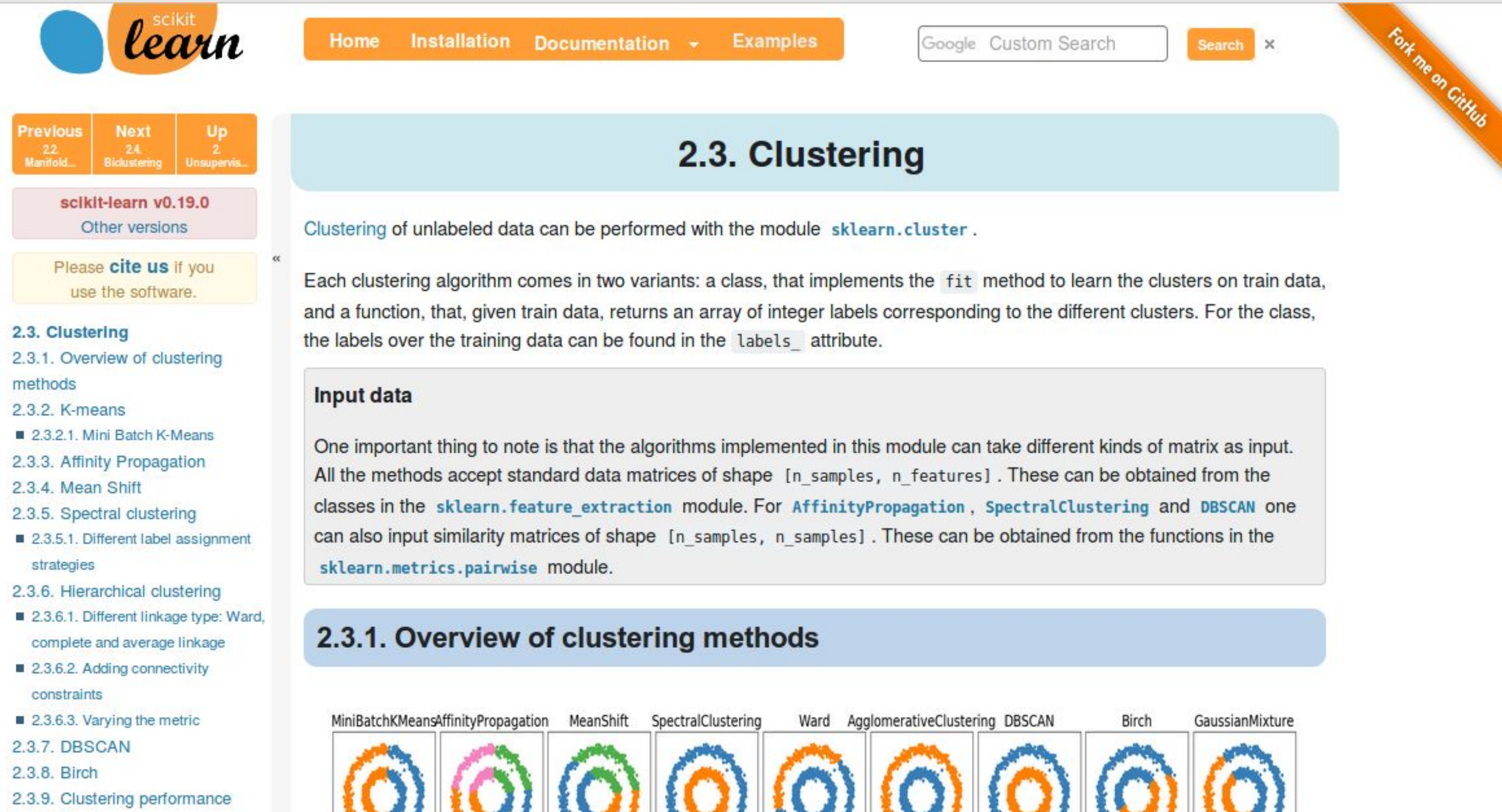
Silhouette Coefficient

- The Silhouette Coefficient s for **a single sample** is given as:

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

- The score is bounded between -1 for incorrect clustering and +1 for highly dense clustering ($a \ll b$). Scores around zero indicate overlapping clusters.

http://scikit-learn.org/stable/modules/clustering.html#clustering



2.3. Clustering

Clustering of unlabeled data can be performed with the module `sklearn.cluster`.

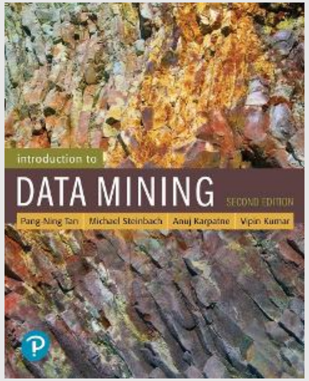
Each clustering algorithm comes in two variants: a class, that implements the `fit` method to learn the clusters on train data, and a function, that, given train data, returns an array of integer labels corresponding to the different clusters. For the class, the labels over the training data can be found in the `labels_` attribute.

Input data

One important thing to note is that the algorithms implemented in this module can take different kinds of matrix as input. All the methods accept standard data matrices of shape `[n_samples, n_features]`. These can be obtained from the classes in the `sklearn.feature_extraction` module. For `AffinityPropagation`, `SpectralClustering` and `DBSCAN` one can also input similarity matrices of shape `[n_samples, n_samples]`. These can be obtained from the functions in the `sklearn.metrics.pairwise` module.

2.3.1. Overview of clustering methods

References



Machine Learning Books

- Pattern Recognition and Machine Learning, Chap. 9 “Mixture Models and EM”
- Pattern Classification, Chap. 10 “Unsupervised Learning and Clustering”
- “Introduction to Data Mining”, https://www-users.cs.umn.edu/~kumar001/dmbook/ch7_clustering.pdf

Machine Learning Courses

- https://www.coursera.org/learn/machine-learning, Week 8