Chair of Process and Data Science

Elements of Machine Learning & Data Science

Clustering

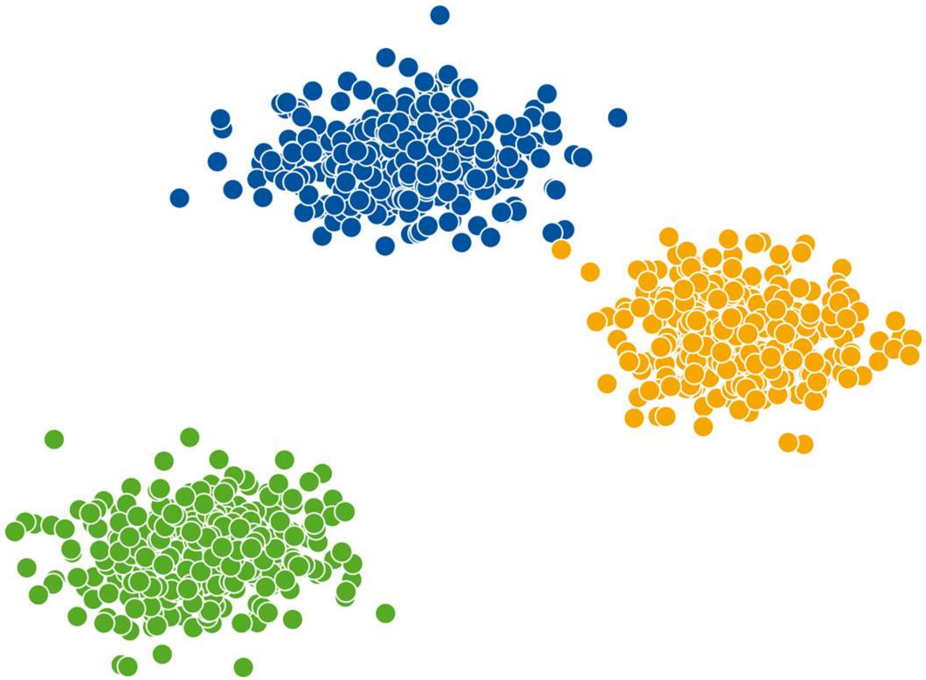
Lecture 8

Prof. Wil van der Aalst

Marco Pegoraro, M.Sc.
Christopher Schwanen, M.Sc.
Tsunghao Huang, M.Sc.

Clustering

1. Introduction to Unsupervised Learning
2. Introduction to Clustering
3. Similarity and Dissimilarity
4. K-means and K-medoids
5. Agglomerative Clustering
6. Density-Based (DBSCAN)
7. Closing



Unsupervised Learning

Unsupervised Learning

- Obtain a model that represents the data…
- …without a target variable or label

features

instances

price	calories	vegetarian	spicy	bestseller
12.99	800	Yes	No	Yes
9.99	600	Yes	Yes	No
14.99	1000	No	Yes	No
11.99	700	No	No	Yes
8.99	500	Yes	No	No

Unsupervised Learning

- Recall from intro lecture: labeled vs unlabeled

features

instances

price	calories	vegetarian	spicy	bestseller
12.99	800	Yes	No	Yes
9.99	600	Yes	Yes	No
14.99	1000	No	Yes	No
11.99	700	No	No	Yes
8.99	500	Yes	No	No

descriptive features

target feature (class label)

instances

price	calories	vegetarian	spicy	bestseller
12.99	800	Yes	No	**Yes**
9.99	600	Yes	Yes	**No**
14.99	1000	No	Yes	**No**
11.99	700	No	No	**Yes**
8.99	500	Yes	No	**No**

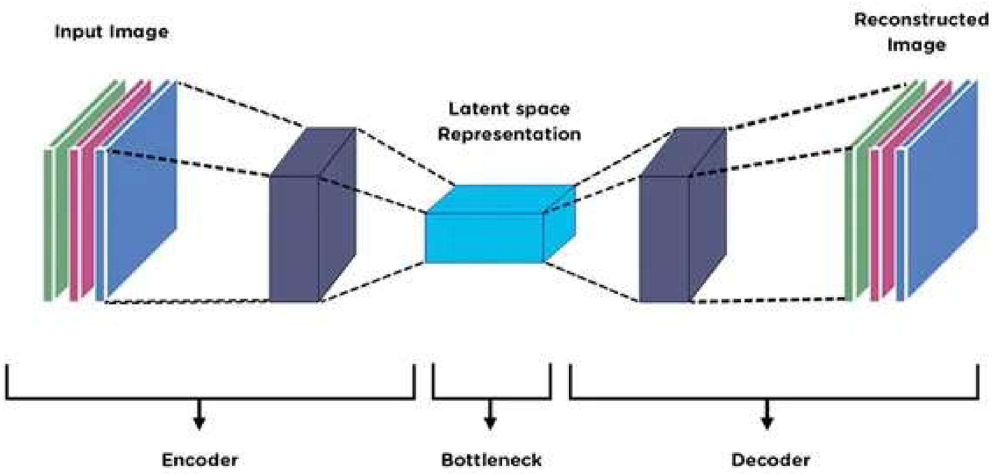
features

Unsupervised Learning

- Obtain a model that represents the data…
- …without a target variable or label

- Why?
  - When a target feature is hard to identify
  - Maybe we are not sure something is even there!
  - To search for patterns in the data
  - To learn a representation

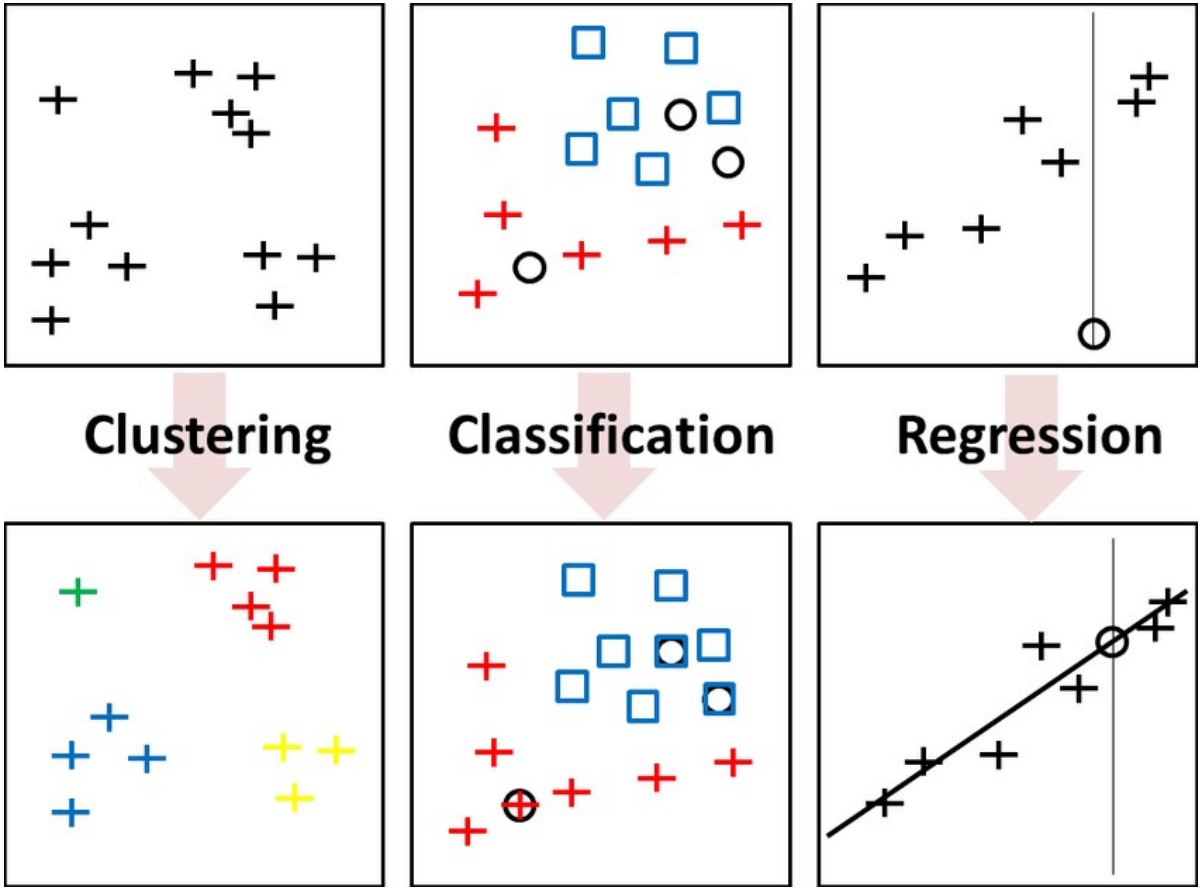
A complex example: autoencoding



MathWorks

Autoencoding: automatically finding a semantics-rich representation of the data in a latent vector space (with the desired dimensionality)

Unsupervised vs. Supervised Learning: Models



F. Nielsen, Parallel Linear Algebra, 2016

In unsupervised learning, the model typically explains relationships between instances

In supervised learning, the model typically explains the values of one or more features

Unsupervised Learning

- Obtain a model that represents the data…
- …without a target variable or label

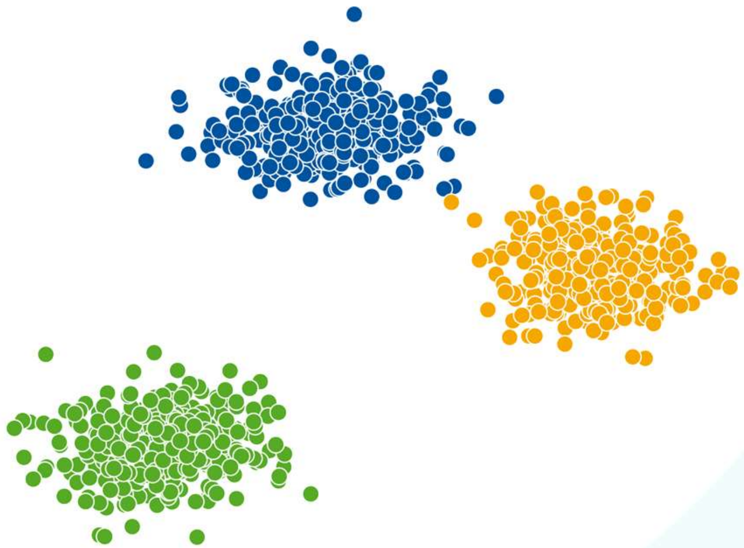
- Challenges:
  - The ground truth might be hard to identify
  - This meaning that designing an evaluation can be very hard

Clustering

Clustering: motivation

- Find clusters (groups of instances) such that:
  - Instances within the cluster are similar
  - Instances in different clusters are dissimilar

- Applications:
  - To find unexpected groups
  - To do data preprocessing
    e.g., discover (process) models for each cluster
  - Unlabeled data is cheaper than labeled data!



Clustering Use Cases

Spotify

User	Song 1	Song 2	Song 3	...
User 1	4	0	5	...
User 2	0	1	0	...
User 3	3	2	9	...
...	...	...	...	...

- 456 million active listeners
- 195 million premium subscribers
- Over 80 million songs

(As of January 2023)

Clustering Use Cases

Amazon

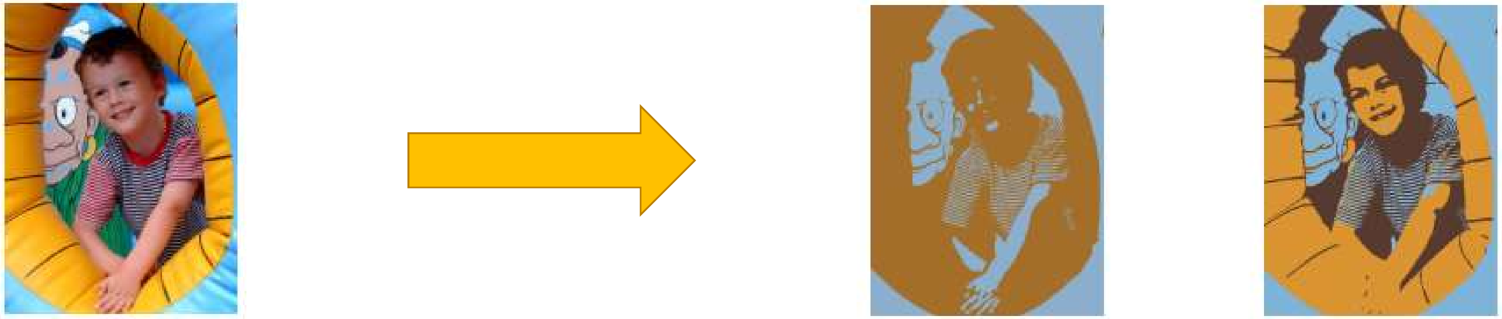
Customer	Prod 1	Prod 2	Prod 3	...
Customer 1	1	0	0	...
Customer 2	0	0	1	...
Customer 3	1	1	0	...
...	...	...	...	...

- 300 million active users
- Over 2 million third-party seller businesses
- Around 350 million items on the marketplace

(As of January 2023)

Clustering Use Cases

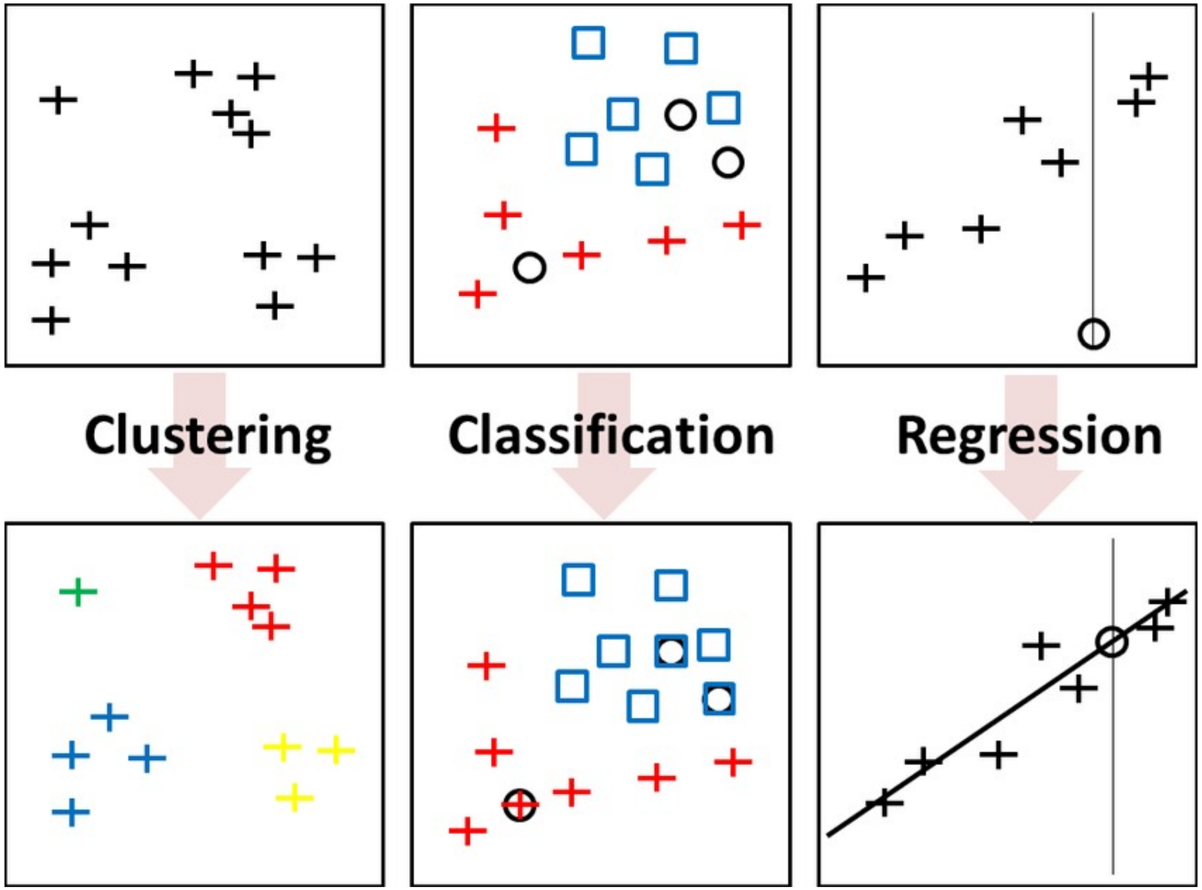
Image Segmentation



C. Bishop, 2006

- Goal: finding regions in a picture with homogeneous appearance
- Applications in computer graphics, e.g. subject detection, edge detection

Clustering, classification and regression



F. Nielsen, Parallel Linear Algebra, 2016

Do not mix up classification with clustering!

When doing classification, we have a training set of correctly classified instances.

When doing clustering, we do not (usually)

Clustering Approaches

- Partitioning methods (split into subsets)
  - Centroid-based (e.g., **k-means**)
  - Medoids-based (e.g., **k-medoids**)
- Hierarchical methods (build dendrogram)
  - **Agglomerative (bottom-up)**
  - Divisive (top-down)
- Density-based methods (e.g., **DBSCAN**)
- Grid-based methods

Similarity and Dissimilarity

Similarity / Dissimilarity

Goal: instances within a cluster are similar, instances in different clusters are dissimilar

$$\mathbf{x_i} = (x_{i1}, x_{i2}, \ldots, x_{iD}) \Longleftrightarrow \mathbf{x_j} = (x_{j1}, x_{j2}, \ldots, x_{jD})$$

Similarity (or proximity)

- Numerical measure of how **alike** two instances are
- **Higher** when instances are more alike
- Often falls in the range [0, 1]

Dissimilarity (or distance)

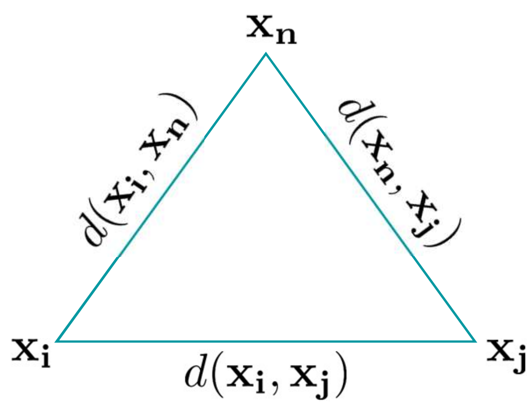
- Numerical measure of how **different** two instances are
- **Lower** when instances are more alike
- Minimum dissimilarity is often 0
- Upper limit varies

inverse

Metric Space Characteristics

$$\mathbf{x_i} = (x_{i1}, x_{i2}, \ldots, x_{iD}) \Longleftrightarrow \mathbf{x_j} = (x_{j1}, x_{j2}, \ldots, x_{jD})$$

- **Non-negativity** - distance is a non-negative number $d(\mathbf{x_i}, \mathbf{x_j}) \geq 0$
- **Identity of indiscernibles** - the distance of an object to itself is 0 $d(\mathbf{x_i}, \mathbf{x_i}) = 0$
- **Symmetry** - distance is a symmetric function $d(\mathbf{x_i}, \mathbf{x_j}) = d(\mathbf{x_j}, \mathbf{x_i})$
- **Triangle inequality** - going directly from object to object in space is no more than going through any other object $d(\mathbf{x_i}, \mathbf{x_j}) \leq d(\mathbf{x_i}, \mathbf{x_n}) + d(\mathbf{x_n}, \mathbf{x_j})$



Examples of Similarity / Dissimilarity Measures

- Binary/Nominal features:
  - Simple matching coefficient
  - Jaccard similarity coefficient
- Continuous features:
  - Euclidean distance $d(\mathbf{x_i}, \mathbf{x_j}) = \sqrt{(x_{i1} - x_{j1})^2 + (x_{i2} - x_{j2})^2 + \cdots + (x_{iD} - x_{jD})^2}$
  - Manhattan distance $d(\mathbf{x_i}, \mathbf{x_j}) = |x_{i1} - x_{j1}| + |x_{i2} - x_{j2}| + \cdots + |x_{iD} - x_{jD}|$
  - Minkowski distance (generalization)
  - Cosine similarity (non-metric measure)

Binary symmetric: Simple matching distance

- Assumes no clear asymmetry between group 0 and 1
- Example: two right-handed persons are as similar as two left-handed persons

	$y = 1$	$y = 0$
$x = 1$	a	b
$x = 0$	c	d

$$SMD(x, y) = \frac{b + c}{a + b + c + d}$$

- a = number of attributes where x and y are both 1
- b = number of attributes where x is 1 and y is 0
- c = number of attributes where x is 0 and y is 1
- d = number of attributes where x and y are both 0

Binary asymmetric: Jaccard distance

- Asymmetry between group 0 and 1
- Example: two persons that won a Turing Award are more similar than two people without a Turing Award

	$y = 1$	$y = 0$
$x = 1$	a	b
$x = 0$	c	d

$$d_J(i,j) = \frac{b + c}{a + b + c}$$

- a = number of attributes where x and y are both 1
- b = number of attributes where x is 1 and y is 0
- c = number of attributes where x is 0 and y is 1
- d = number of attributes where x and y are both 0

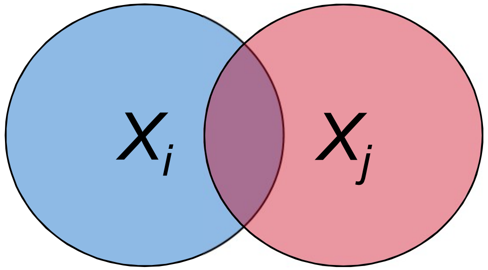
Nominal: Simple matching distance

- mm: total number of variables where objects i and j mismatch
- p: total number of variables

$$SMD(i,j) = \frac{mm}{p}$$

- Simple matching coefficient: $SMC = 1 - SMD$

Nominal: Jaccard Similarity Coefficient



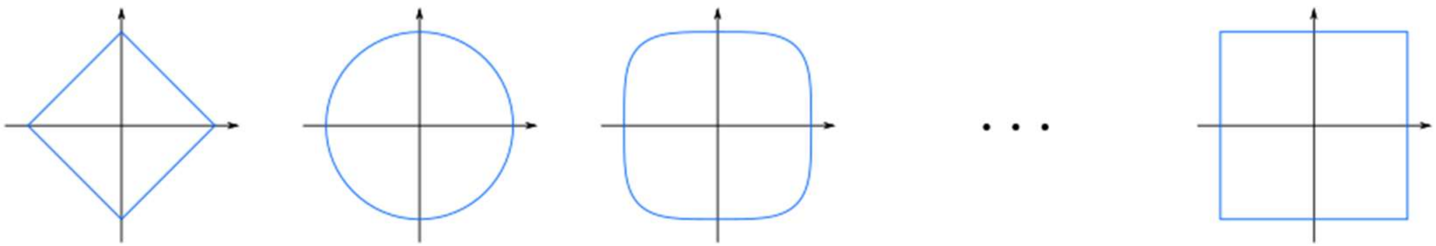
- Assumes instances are represented by sets
- Jaccard similarity between two sets X_i and X_j
  $J(X_i, X_j) = \frac{|X_i \cap X_j|}{|X_i \cup X_j|}$
- Jaccard distance between two sets X_i and X_j
  $d_J(X_i, X_j) = \frac{|X_i \cup X_j| - |X_i \cap X_j|}{|X_i \cup X_j|} = 1 - J(X_i, X_j)$
- Jaccard distance is a metric, i.e., distance is non-negative, distance to itself is zero, symmetric, and satisfies the triangle inequality
- Used for comparing item sets (e.g., products ordered, words appearing in documents, courses taken, and songs played)

Minkowski Distance (Metric)

Generalization of Manhattan and Euclidean distance to any natural dimension $p \geq 1$ (also called L^p norm)

$$d(\mathbf{x_i}, \mathbf{x_j}) = \sqrt[p]{|x_{i1} - x_{j1}|^p + |x_{i2} - x_{j2}|^p + \cdots + |x_{iD} - x_{jD}|^p}$$

Manhattan Euclidean Maximum/Chebyshev distance



$p = 2^0 = 1$ $p = 2^1 = 2$ $p = 2^2 = 4$ $p = 2^\infty = \infty$ [1]

Minkowski Distance – Examples

$$d(\mathbf{x_i}, \mathbf{x_j}) = \sqrt[p]{|x_{i1} - x_{j1}|^p + |x_{i2} - x_{j2}|^p + \cdots + |x_{iD} - x_{jD}|^p}$$

Manhattan distance $p = 1$

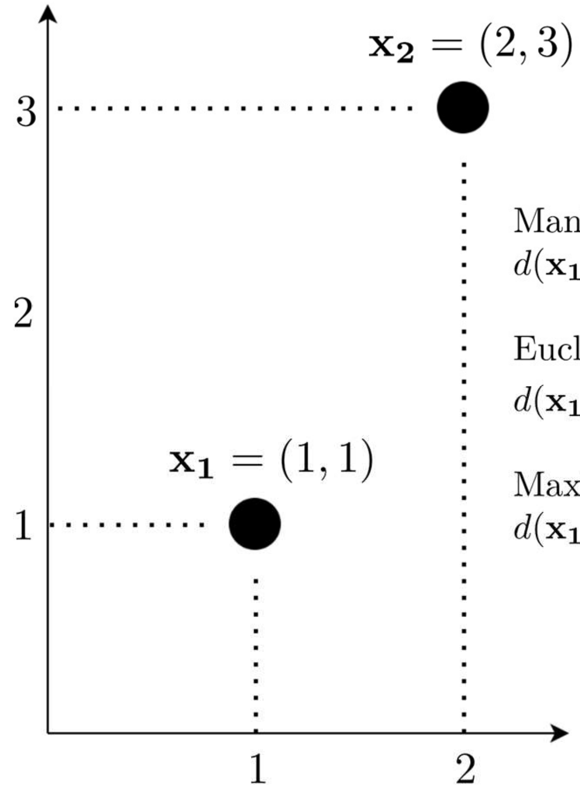
$$d(\mathbf{x_i}, \mathbf{x_j}) = |x_{i1} - x_{j1}| + |x_{i2} - x_{j2}| + \cdots + |x_{iD} - x_{jD}|$$

Euclidean distance $p = 2$

$$d(\mathbf{x_i}, \mathbf{x_j}) = \sqrt{(x_{i1} - x_{j1})^2 + (x_{i2} - x_{j2})^2 + \cdots + (x_{iD} - x_{jD})^2}$$

Maximum/Chebyshev distance $p \to \infty$

$$d(\mathbf{x_i}, \mathbf{x_j}) = \lim_{p\to\infty} \left(\sum_{d=1}^{D} |x_{id} - x_{jd}|^p \right)^{\frac{1}{p}} = \max_{d\in\{1,\ldots,D\}} |x_{id} - x_{jd}|$$



Manhattan distance
$d(\mathbf{x_1}, \mathbf{x_2}) = |1-2| + |1-3| = 3$

Euclidean distance
$d(\mathbf{x_1}, \mathbf{x_2}) = \sqrt{(1-2)^2 + (1-3)^2} = \sqrt{5}$

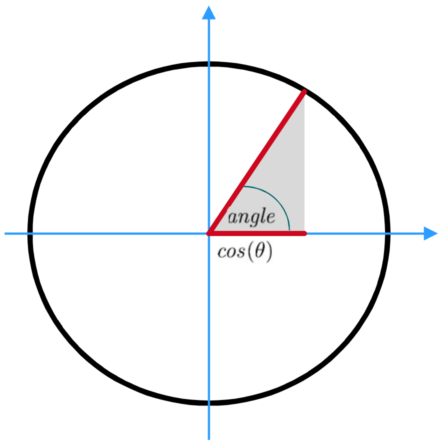
Maximum/Chebyshev distance
$d(\mathbf{x_1}, \mathbf{x_2}) = max\{|3-1|, |2-1|\} = 2$

Cosine Similarity (Non-Metric)

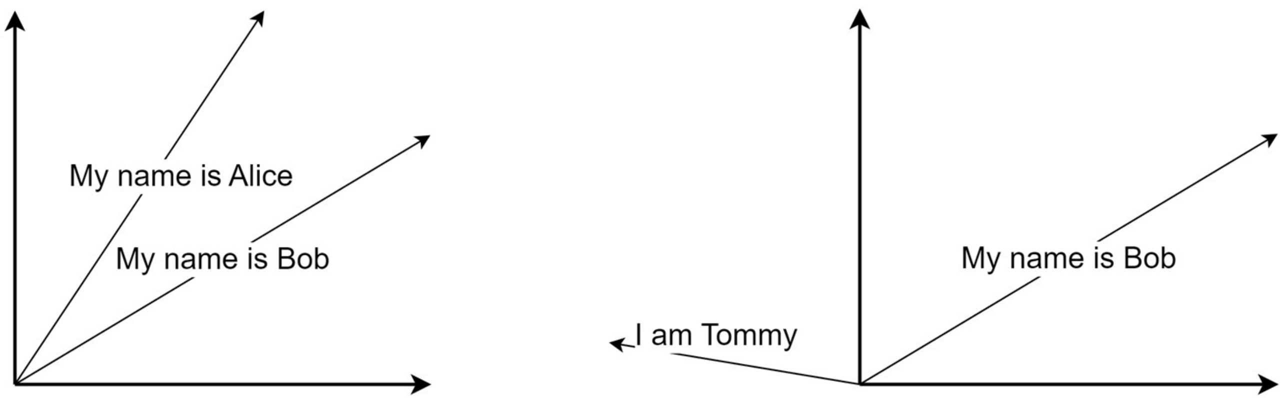
- Cosine similarity between two vectors $\mathbf{v}$ and $\mathbf{w}$

$$S_C(\mathbf{x_i}, \mathbf{x_j}) = cos(\theta) = \frac{\mathbf{x_i} \cdot \mathbf{x_j}}{||\mathbf{x_i}||\,||\mathbf{x_j}||}$$

- Used for sparse data (focus on angle rather than distance)
- Often used for comparing representations of textual data



angle	cos(θ)
0°	1.0000
45°	0.7071
90°	0.0000
135°	-0.7071
180°	-1.0000
270°	0.0000
360°	1.0000



Similar vectors
Positive, approaches 1.0

Different vectors
Negative, approaches -1.0

Mixing Different Types of Features

- Commonly used approach – normalize all features ranges to [0, 1]
- One can give different weights to different features (i.e., distances are not the same in all dimensions)
- One can exclude features, because they would lead to obvious clusters (providing no insights)

K-means and K-medoids

K-means – Idea

- Let's start with very strong assumptions
  - And therefore, a simpler problem
- What if we could represent a cluster with a single point in space?

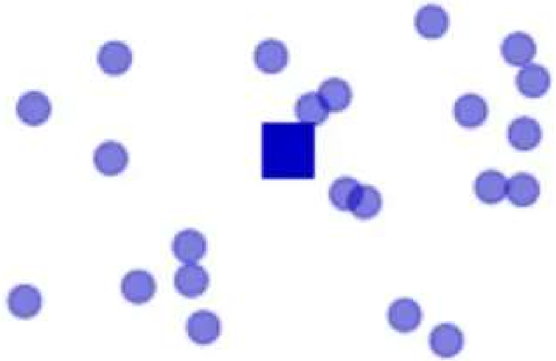
- Then, grouping the instances in clusters would be very easy:
- Given an instance, find the closest representative, and assign the corresponding cluster

K-means – Idea

- Let's consider the opposite problem:
  From the set of instances that we assume to belong to a cluster, how do we find this representative?

- A natural choice is to pick the point at the geometric center of the instances
- Or, equivalently, the center of the smallest sphere that contains all the instances
  - (given a certain distance metric)
- Also easy!

- Let's call this representative a **centroid**



K-means – Idea

- But then…
  - Finding the instances in a cluster given its centroid is easy
  - Finding a centroid given the instances in its cluster is also easy
  - But we have neither!
  - A chicken and egg problem!

K-means – Idea

- But then…
  - Finding the instances in a cluster given its centroid is easy
  - Finding a centroid given the instances in its cluster is also easy
  - But we have neither!
  - A chicken and egg problem
- Solution: we start with random points in space as centroids, and we iteratively refine

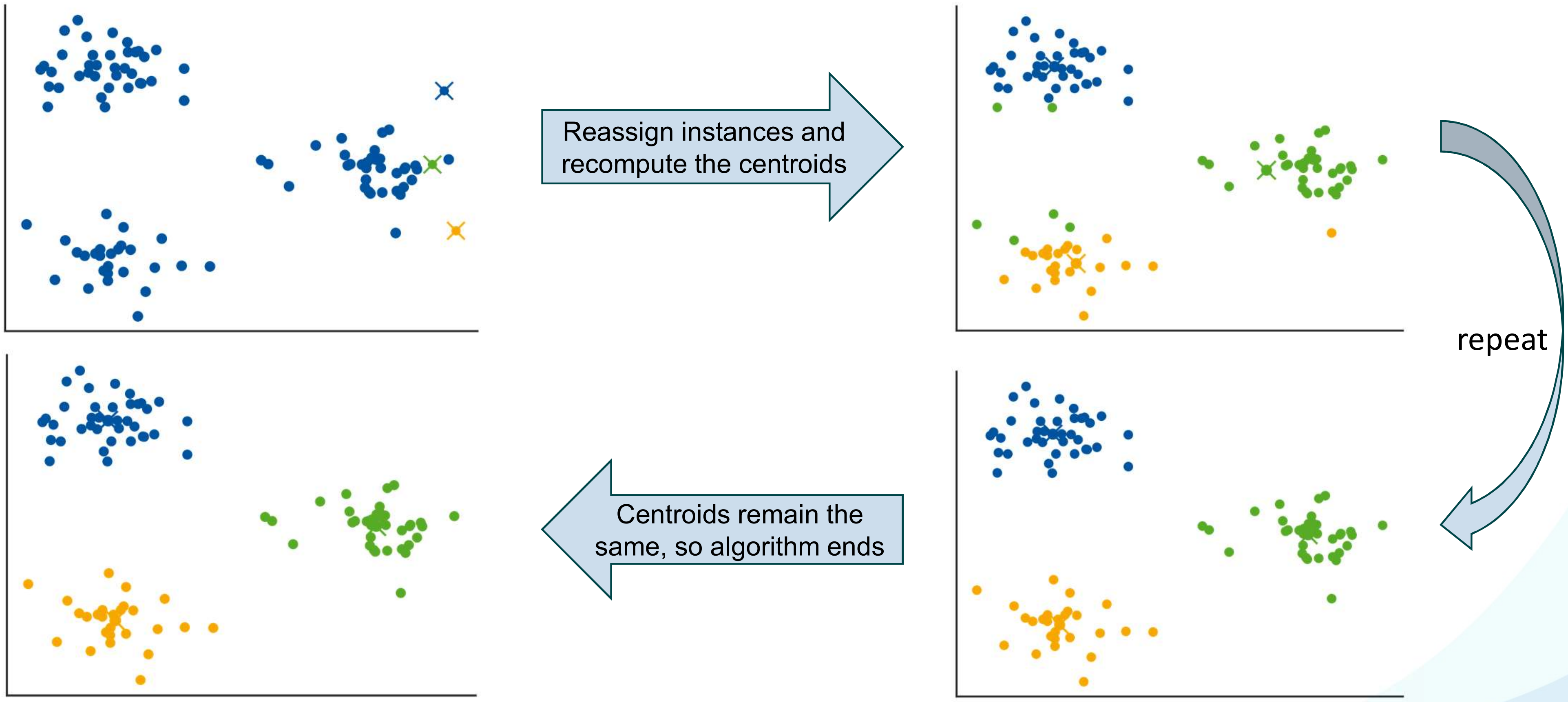
K-means

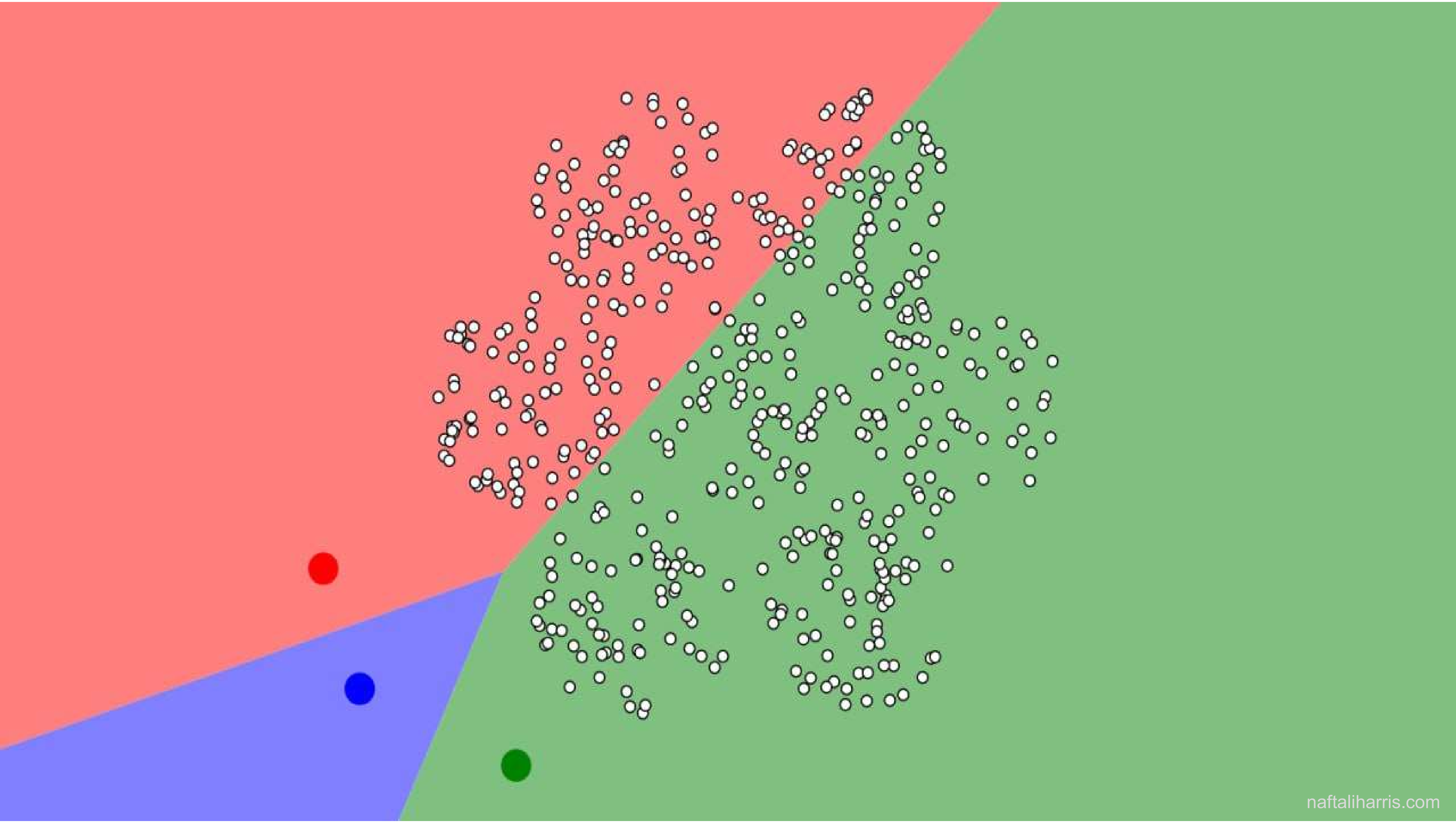
- Algorithm for clustering / partitioning data
- Each cluster's center (the **centroid**) is represented by the mean value of the instances (points) in the cluster
- Simple and fast to compute

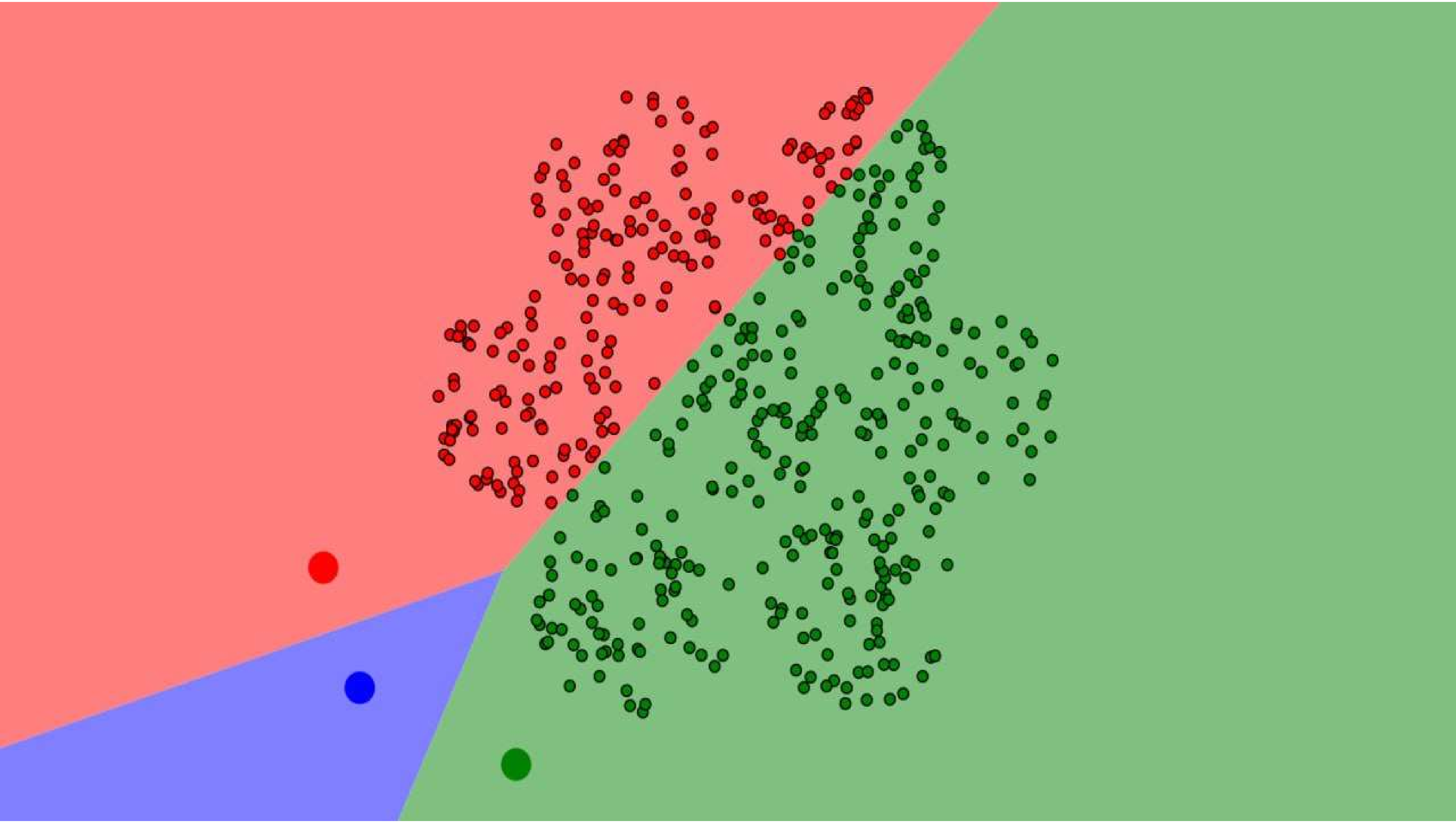
K-means algorithm:

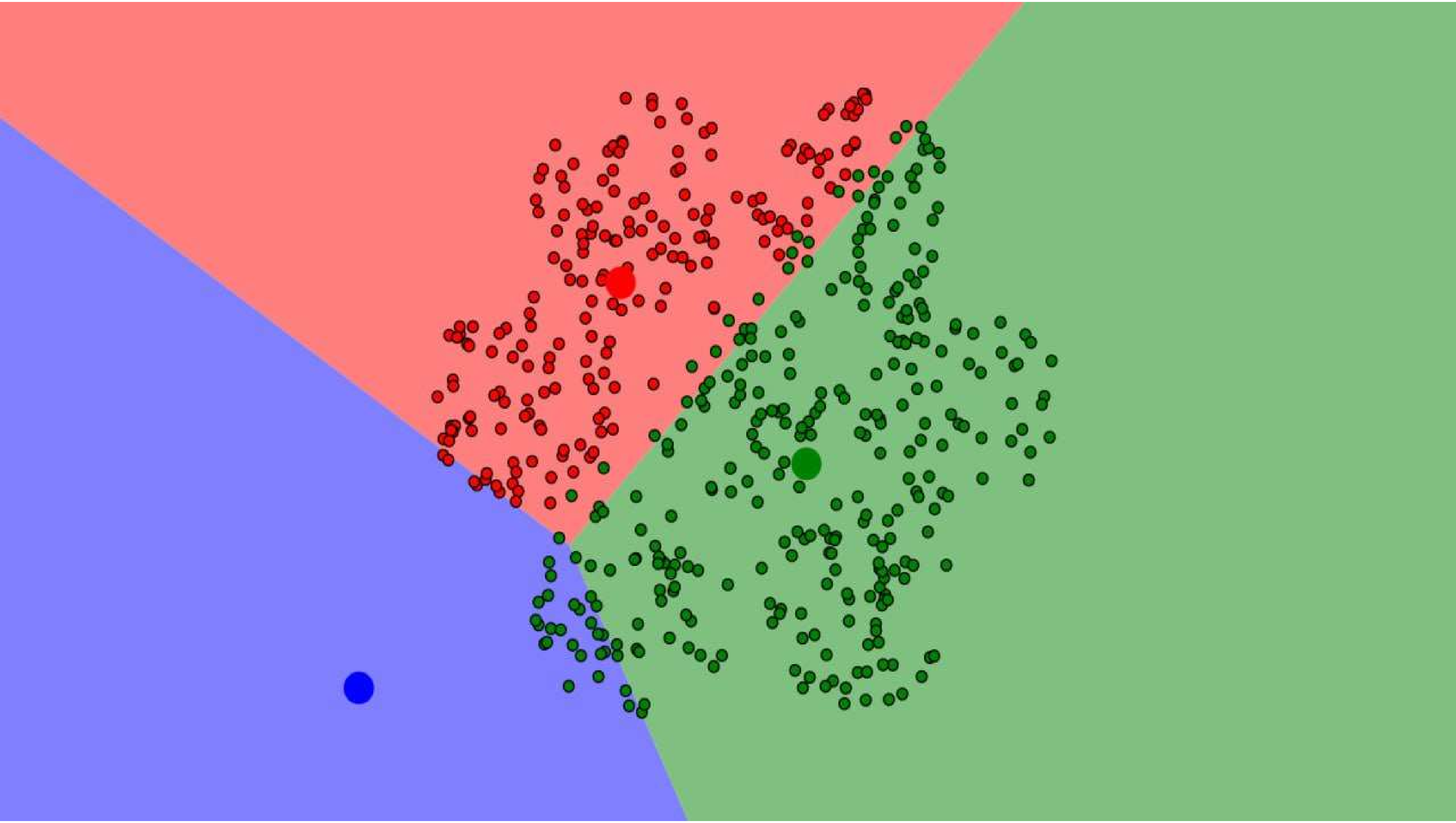
1. randomly choose k instances from the dataset $\mathcal{X}$ as the initial cluster centers
2. **repeat until no change**
   (a) reassign each instance to the cluster with the closest centroid
   (b) recompute the centroid $\mathbf{c_i}$ for each cluster $\mathcal{C}_i$ for $i = 1, \ldots, k$

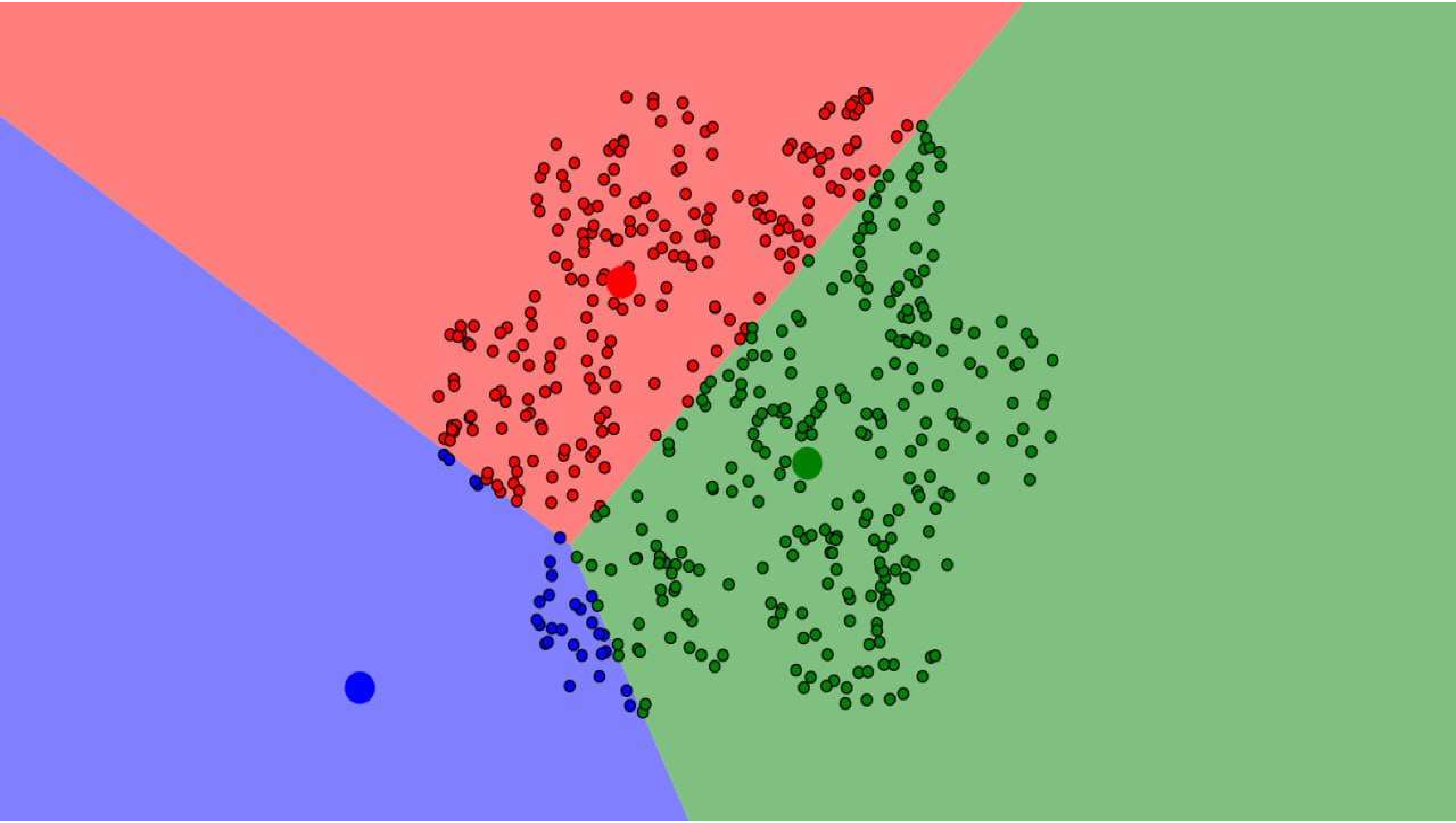
K-means – Example

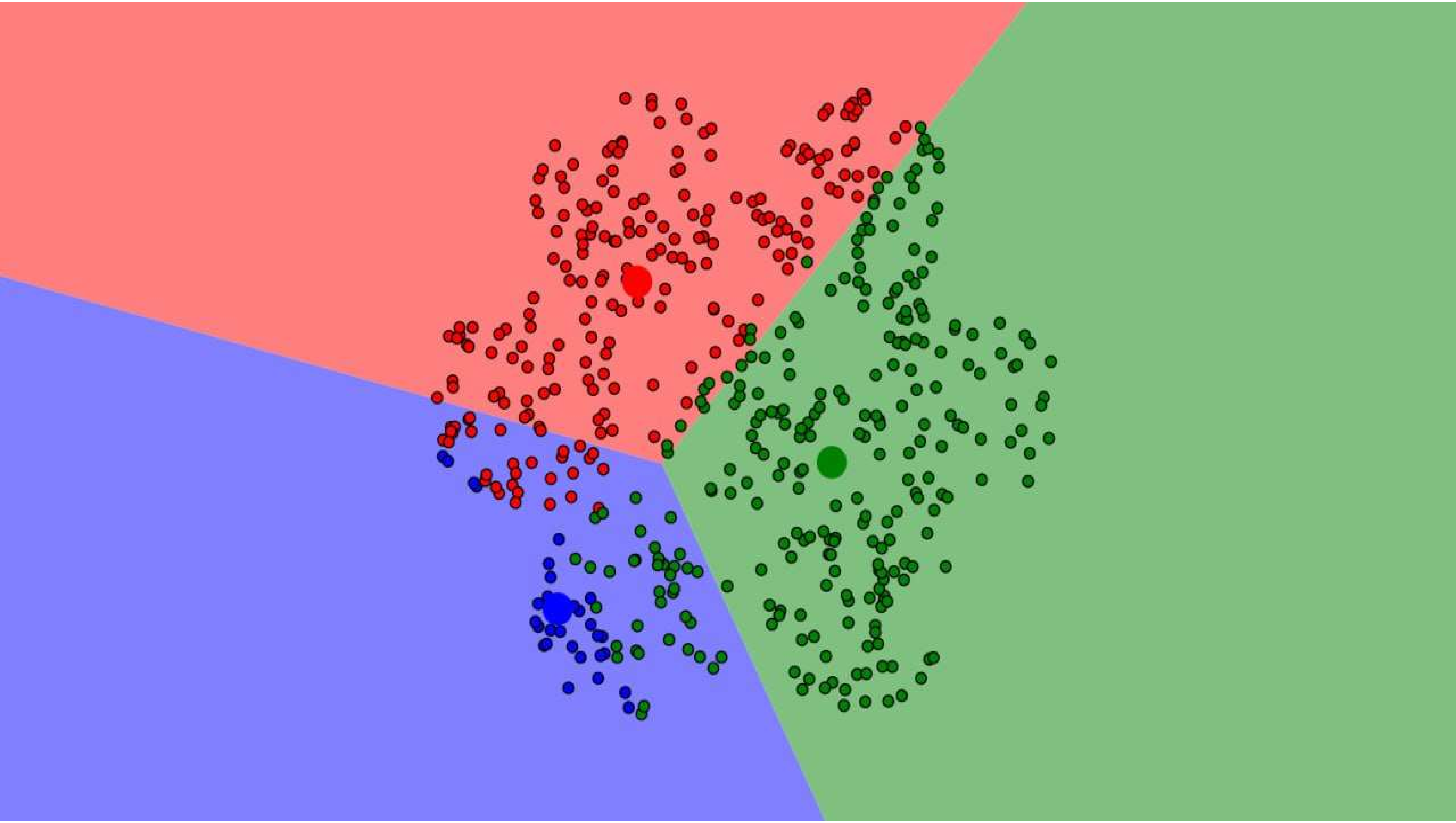


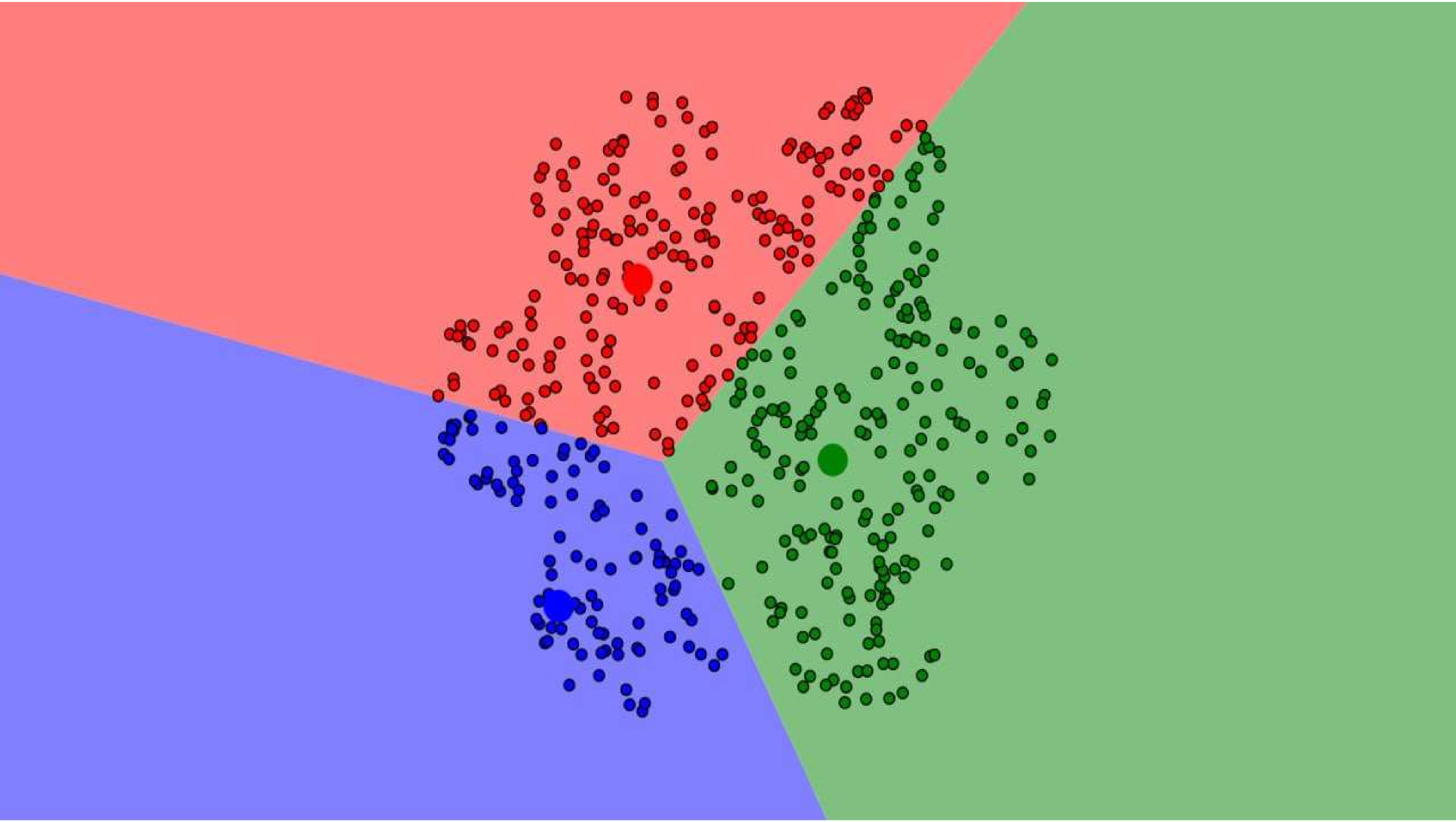

naftaliharris.com

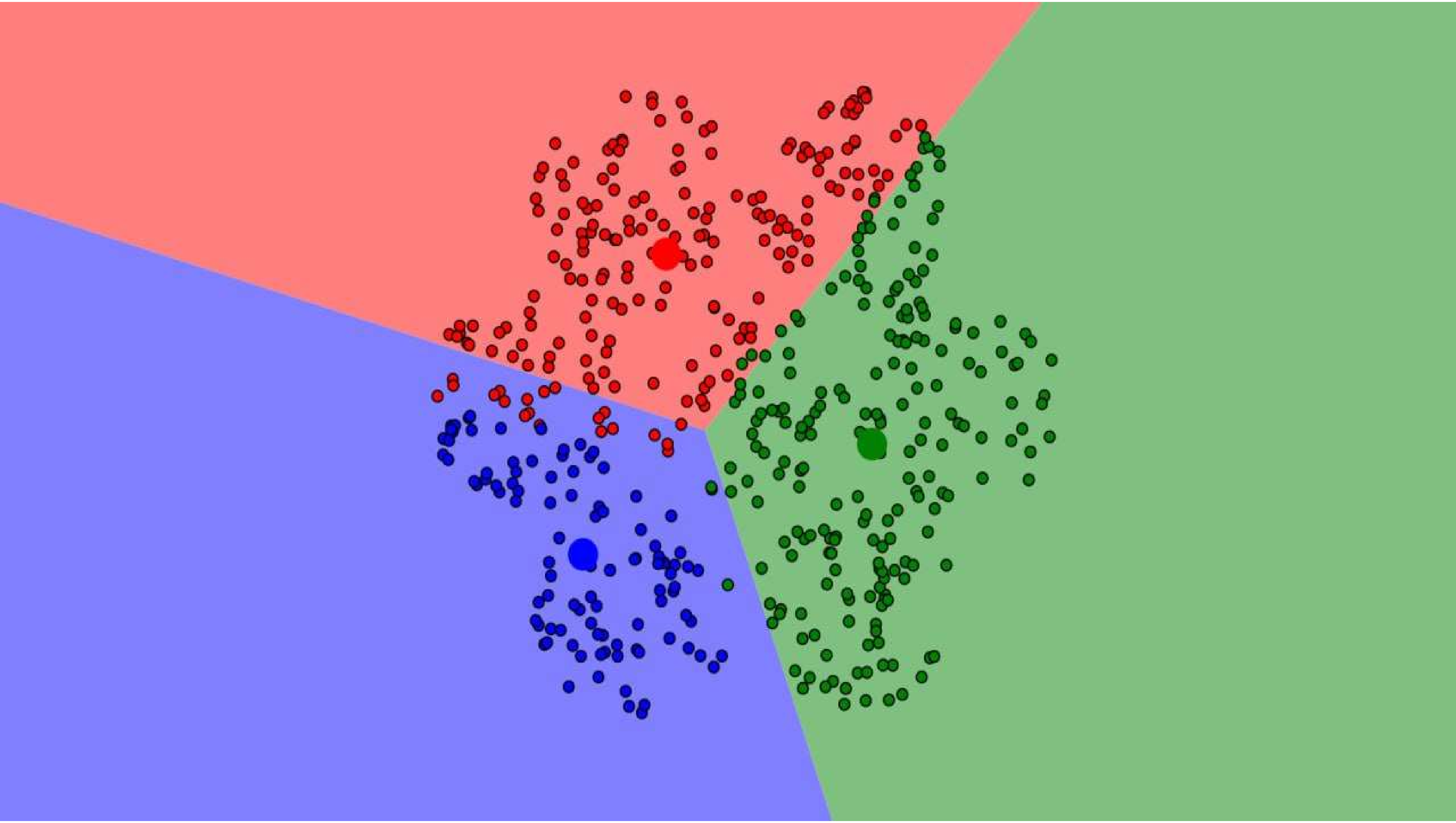


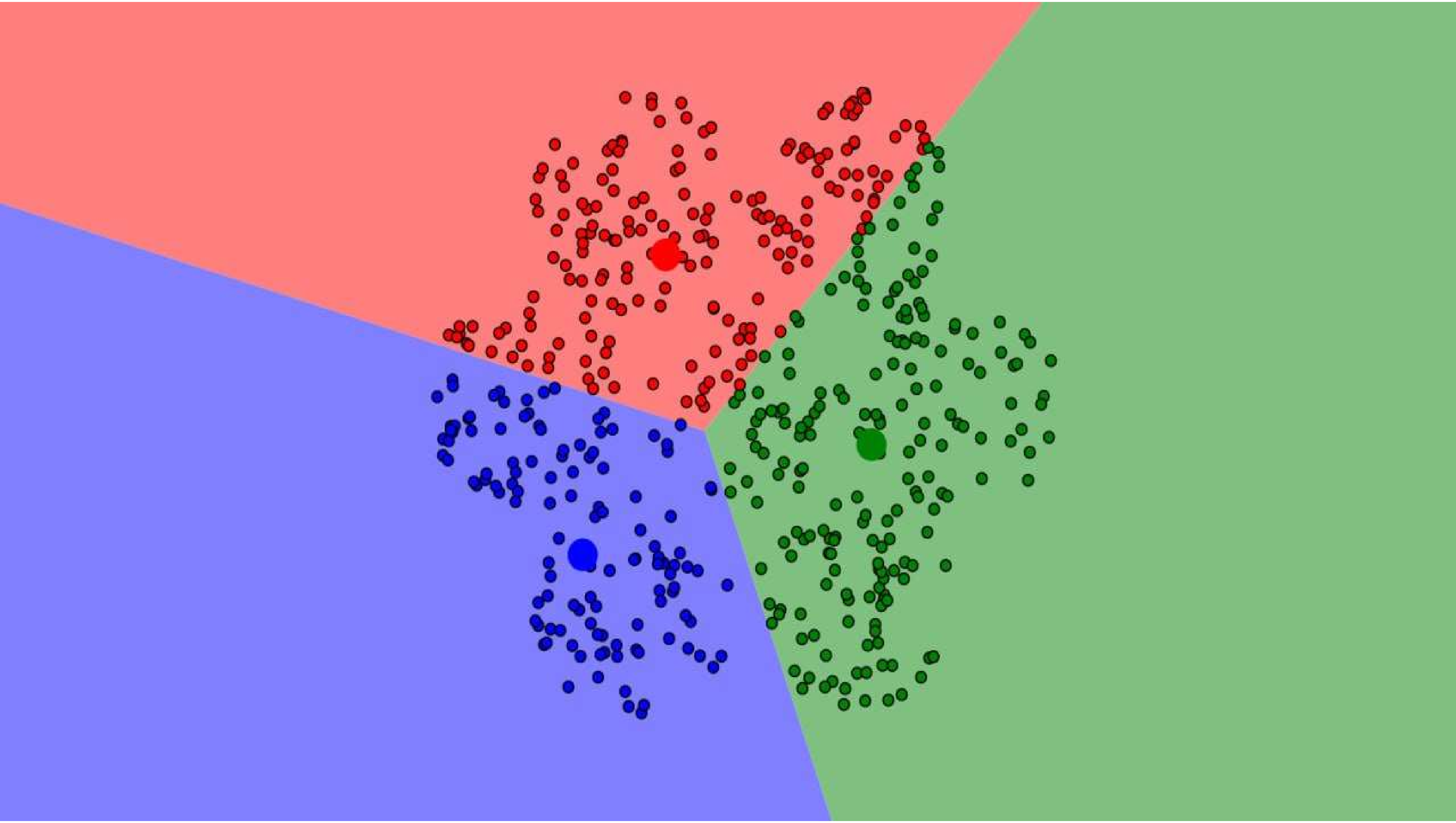


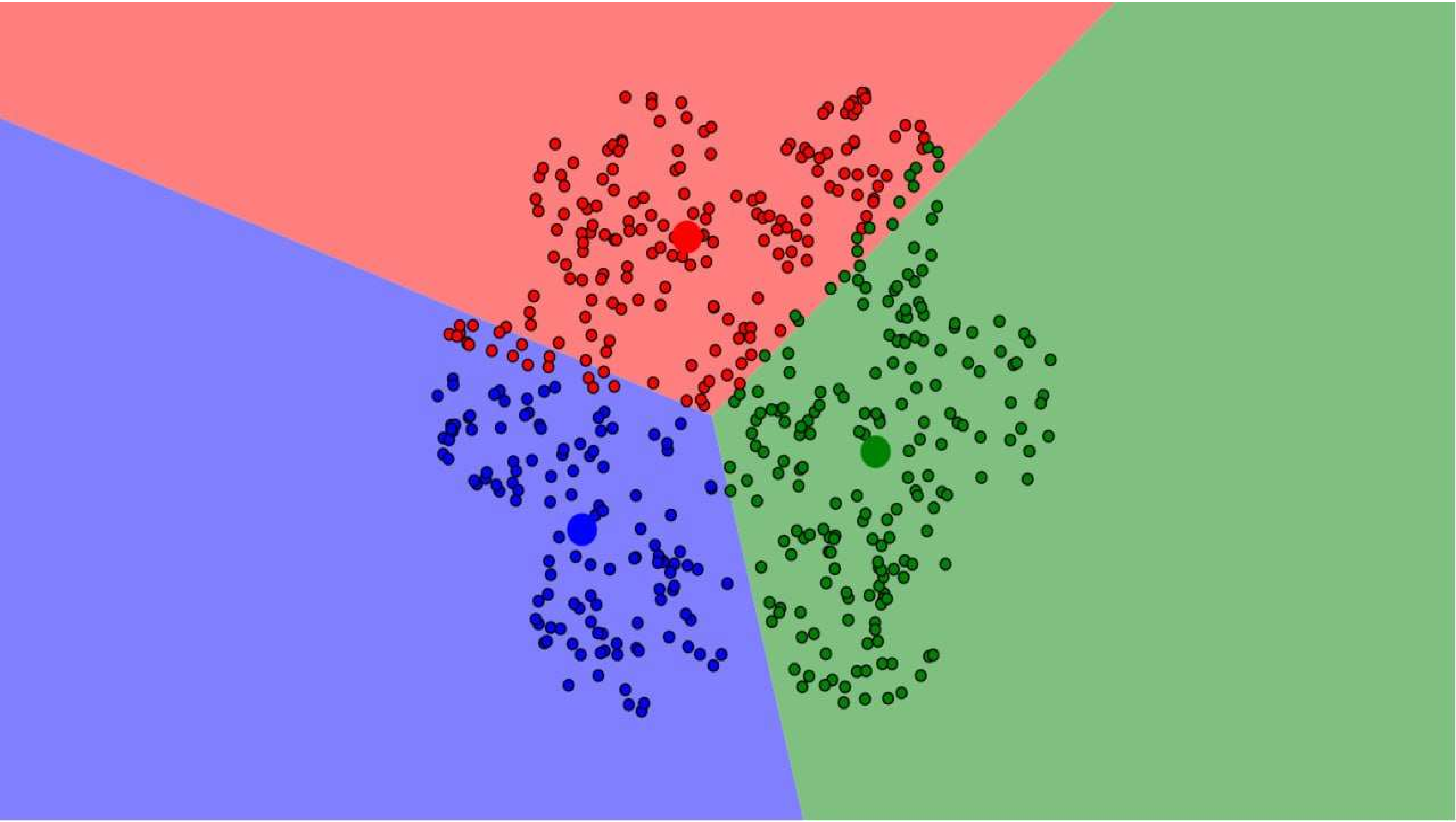


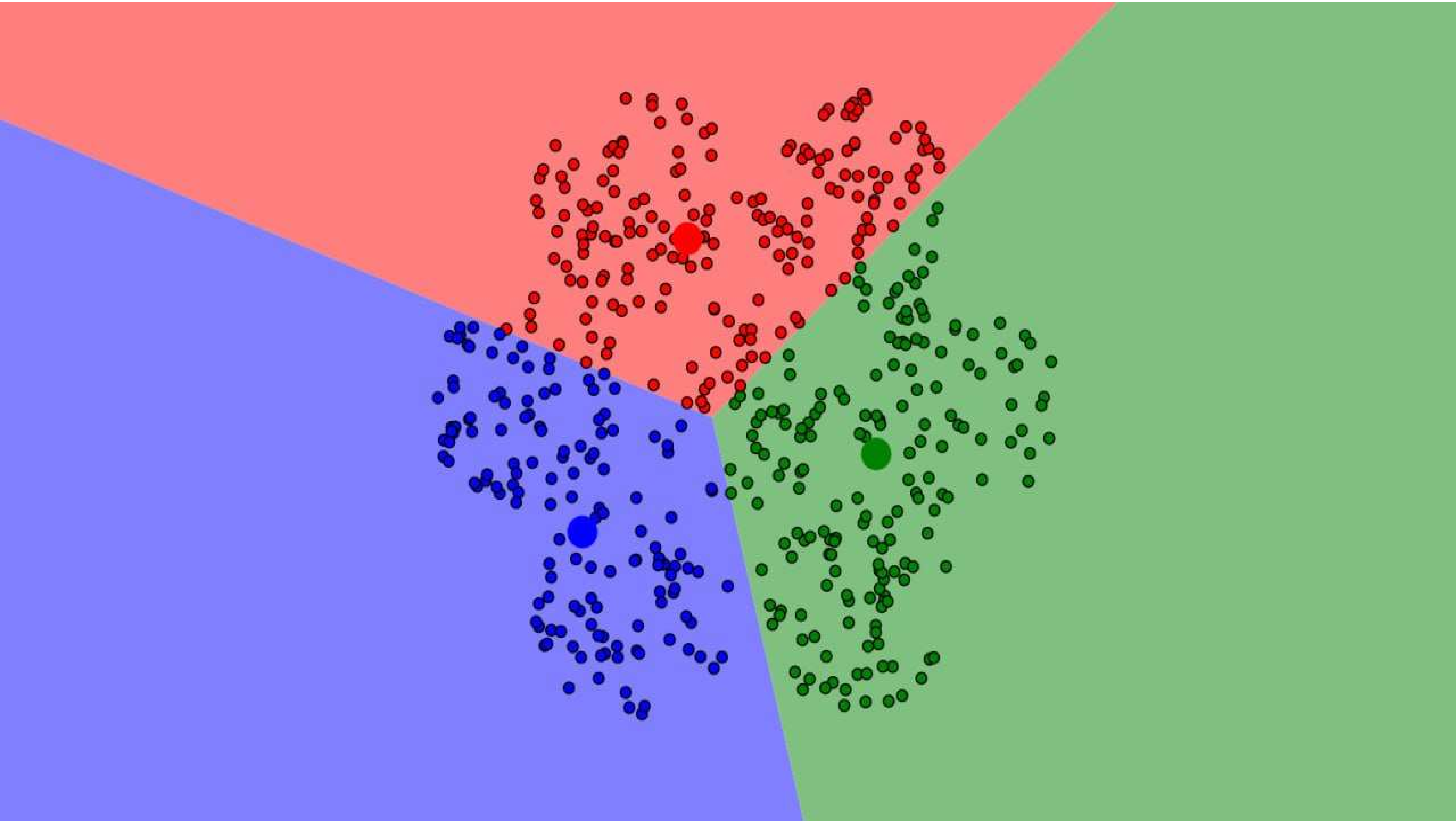








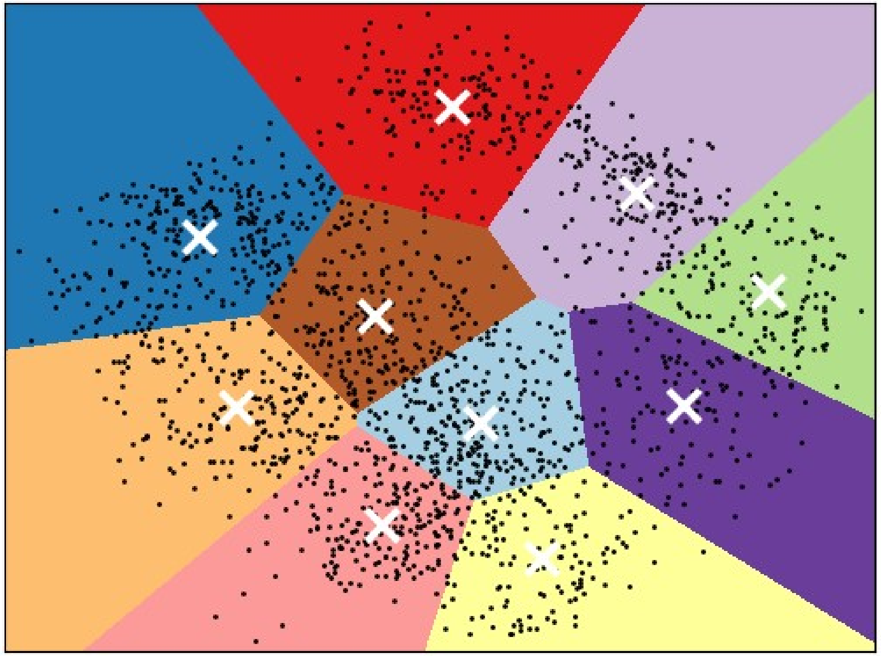




K-means – Example

- Note: even though you can think of K-means clusters as (truncated) spheres enclosing instances, the final goal is to partition the instance space:

J. Mx



Scikit-learn

K-means – Quality of Clusters

Error is typically described as the sum of all squared errors between all instances and their closest centroids

Number of clusters

Instance x_j in cluster C_i

Centroid of cluster C_i

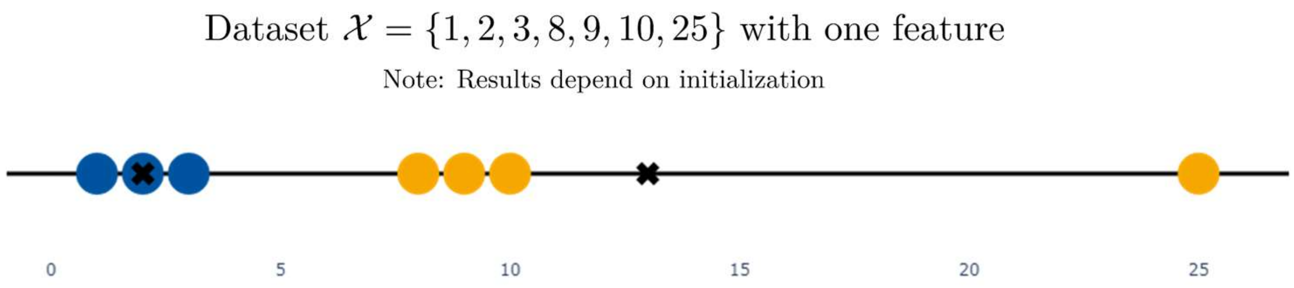
$$Error = \sum_{i=1}^{k} \sum_{\mathbf{x_j} \in \mathcal{C}_i} dist\left(\mathbf{x_j}, \mathbf{c_i}\right)^2$$

Image Segmentation with K-means

C. Bishop, 2006

K-means – Limitations

- Number k and the distance metric need to be chosen beforehand
- Assumes that clusters have spherical shape and similar density
- Different initial points often lead to different results (in practice k-means is run multiple times to minimize this problem)
- Sensitive to outliers

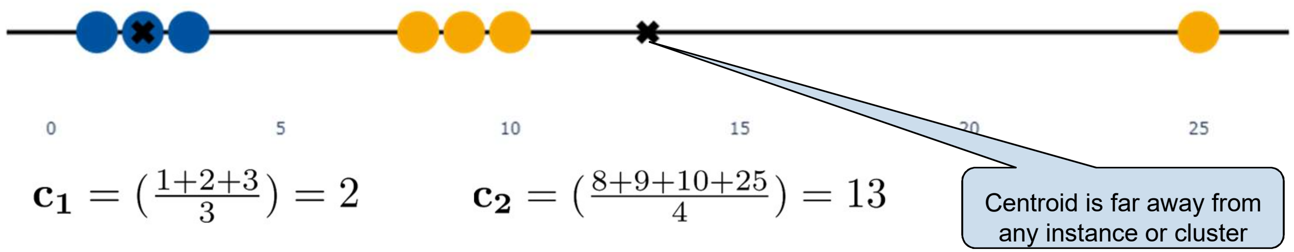


K-means – Limitations

- Number k and the distance metric need to be chosen beforehand
- Assumes that clusters have spherical shape and similar density
- Different initial points often lead to different results (in practice k-means is run multiple times to minimize this problem)
- Sensitive to outliers

Dataset $\mathcal{X} = \{1, 2, 3, 8, 9, 10, 25\}$ with one feature

Note: Results depend on initialization



$$\mathbf{c_1} = \left(\frac{1+2+3}{3}\right) = 2 \qquad \mathbf{c_2} = \left(\frac{8+9+10+25}{4}\right) = 13$$

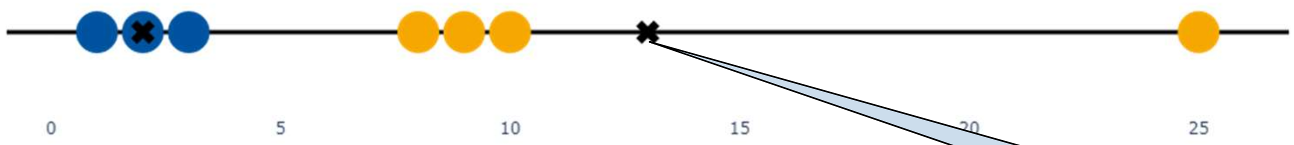
Algorithm terminates because each instance is assigned to correct centroid

K-means – Limitations

- Number k and the distance metric need to be chosen beforehand
- Assumes that clusters have spherical shape and similar density
- Different initial points often lead to different results (in practice k-means is run multiple times to minimize this problem)
- Sensitive to outliers

Dataset $\mathcal{X} = \{1, 2, 3, 8, 9, 10, 25\}$ with one feature

Note: Results depend on initialization



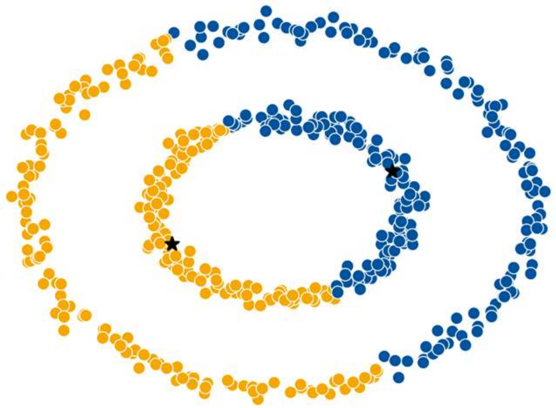
$\mathbf{c_1} = \left(\frac{1+2+3}{3}\right) = 2 \qquad \mathbf{c_2} = \left(\frac{8+9+10+25}{4}\right) = 13$

Algorithm terminates because each instance is assigned to correct centroid

$$
\begin{aligned}
Error_{C_1} &= (1-2)^2 + (2-2)^2 + (3-2)^2 = 2 \\
Error_{C_2} &= (8-13)^2 + (9-13)^2 + (10-13)^2 + (25-13)^2 = 194 \\
Error &= Error_{C_1} + Error_{C_2} = 196
\end{aligned}
$$

K-means – Limitations

- Number k and the distance metric need to be chosen beforehand
- Assumes that clusters have spherical shape and similar density
- Different initial points often lead to different results (in practice k-means is run multiple times to minimize this problem)
- Sensitive to outliers



K-medoids – Idea

- Uses concrete instances (**medoids**) as cluster's centers rather than the mean values (centroids)
- Similar idea to K-means
- Error is again based on the distances

Number of clusters

Instance $\boldsymbol{x_j}$ in cluster $\boldsymbol{C_i}$

Medoid of cluster $\boldsymbol{C_i}$

$$Error = \sum_{i=1}^{k} \sum_{\mathbf{x_j} \in \mathcal{C}_i} dist\left(\mathbf{x_j}, \mathbf{m_i}\right)^2$$

K-medoids – Algorithm

- Uses concrete instances (**medoids**) as cluster's centers rather than the mean values (centroids)
- In literature medoids are also known as representative instances

K-medoids algorithm:

1. randomly choose k instances from the dataset $\mathcal{X}$ as the initial cluster centers
2. **repeat until no change**
   (a) reassign each instance to the cluster with the closest medoid
   (b) for each medoid $\mathbf{m_i}$ and each non-medoid instance $\mathbf{x_j}$
      i. compute the error for the clustering assuming that we **swap** medoid $\mathbf{m_i}$ by $\mathbf{x_j}$
      ii. if the error is lower, perform the **swap**

Comparing K-medoids and K-means

- More robust to outliers (e.g., 1D example on the right)
- K-medoids is more flexible (can be used with any similarity measure)
- K-medoids is more time-consuming (although the effect of swaps is limited to the instances that change medoid)

Dataset $\mathcal{X} = \{1, 2, 3, 8, 9, 10, 25\}$ with one feature

Note: Results depend on initialization

Algorithm terminates because there is no swap that lowers the error

Comparing K-medoids and K-means

- More robust to outliers (e.g., 1D example on the right)
- K-medoids is more flexible (can be used with any similarity measure)
- K-medoids is more time-consuming (although the effect of swaps is limited to the instances that change medoid)

Dataset $\mathcal{X} = \{1, 2, 3, 8, 9, 10, 25\}$ with one feature

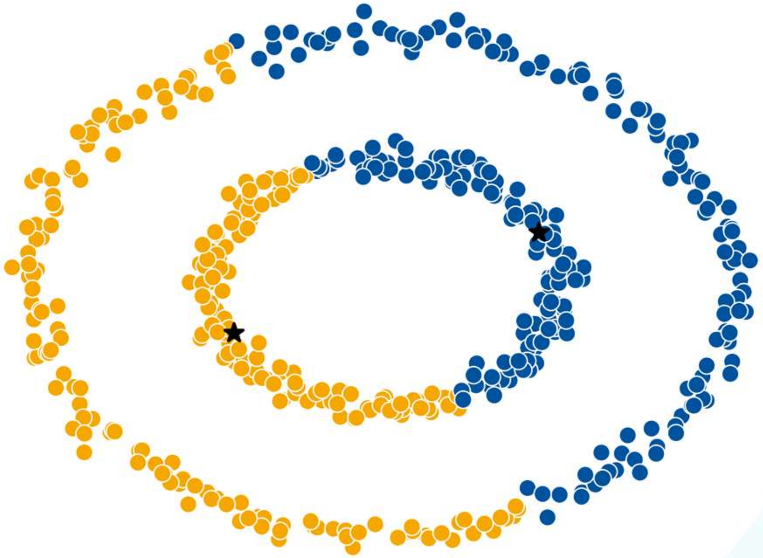
Note: Results depend on initialization

Algorithm terminates because there is no swap that lowers the error

$$Error_{C_1} = (1-3)^2 + (2-3)^2 + (3-3)^2 + (8-3)^2 + (9-3)^2 + (10-3)^2 = 115$$
$$Error_{C_2} = (25-25)^2 = 0$$
$$Error = Error_{C_1} + Error_{C_2} = 115$$

K-means error was 196

K-means and K-medoids – Shape Limitations



K-means and K-medoids – Choosing K

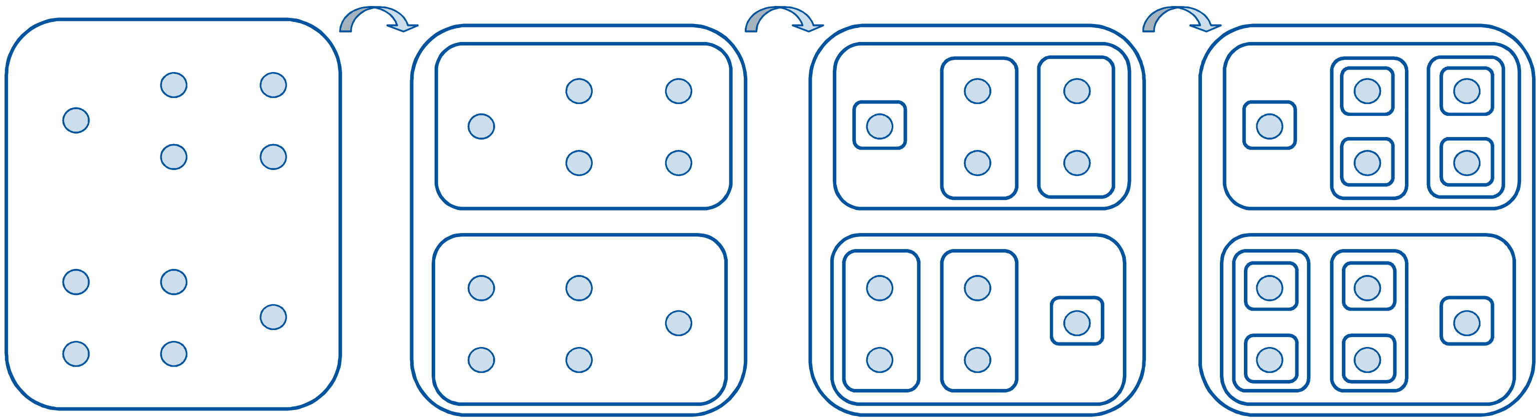
- The choice of a good value for K is quite hard!

- Connects with the more general issue of evaluation of unsupervised learning approaches

- Some ideas:
  - Domain knowledge: the guidance of the data owners is important!
  - Random restart: we perform clustering multiple times with multiple Ks, we keep the best
  - Holdout: we split the dataset, we test various Ks on part of the data and measure the error on another
  - Bayesian: sometimes, we may have a prior on values of K

Agglomerative Clustering

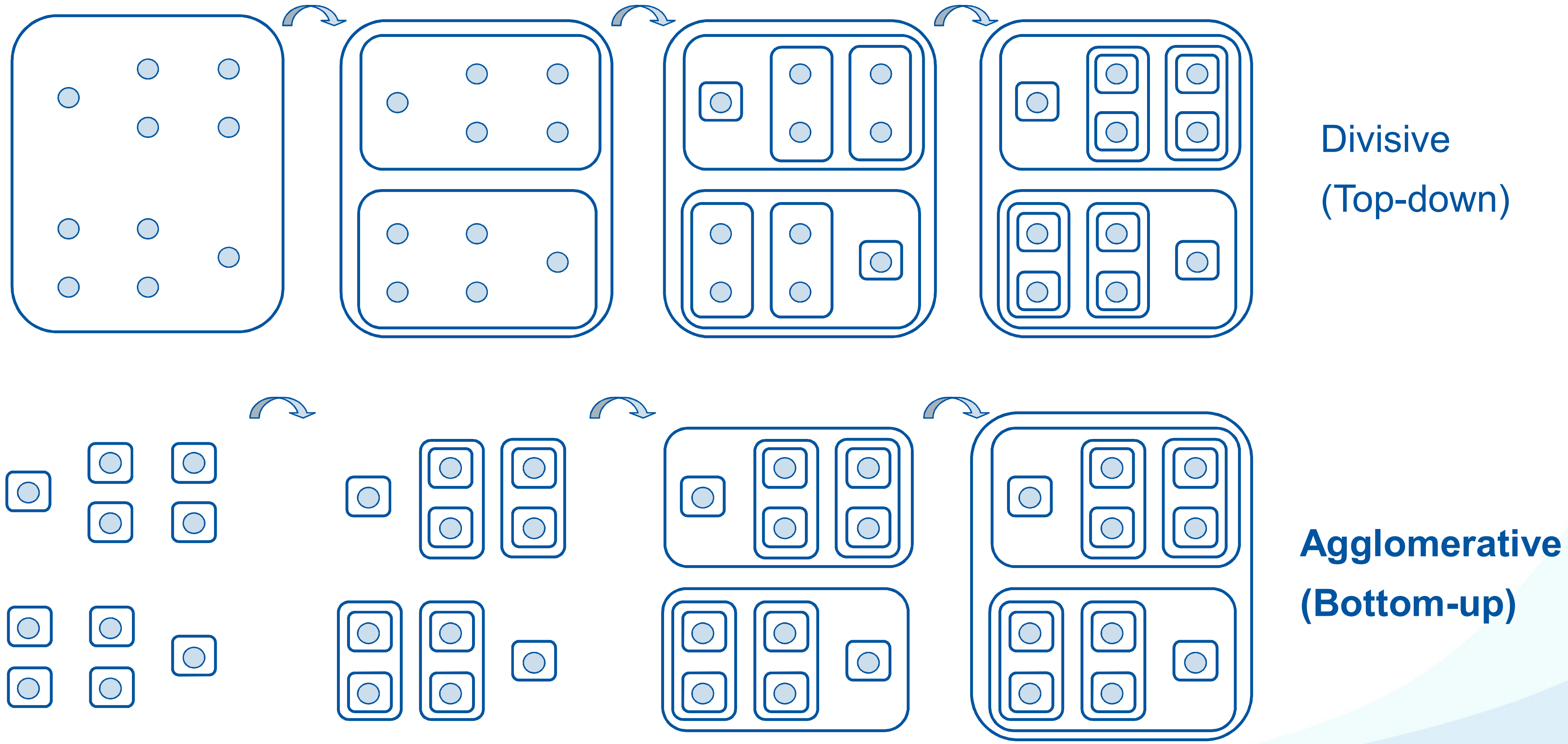
Hierarchical Clustering



Divisive

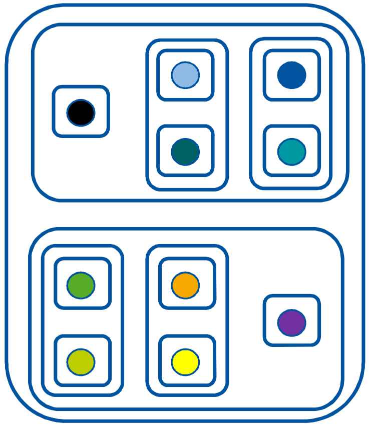
(Top-down)

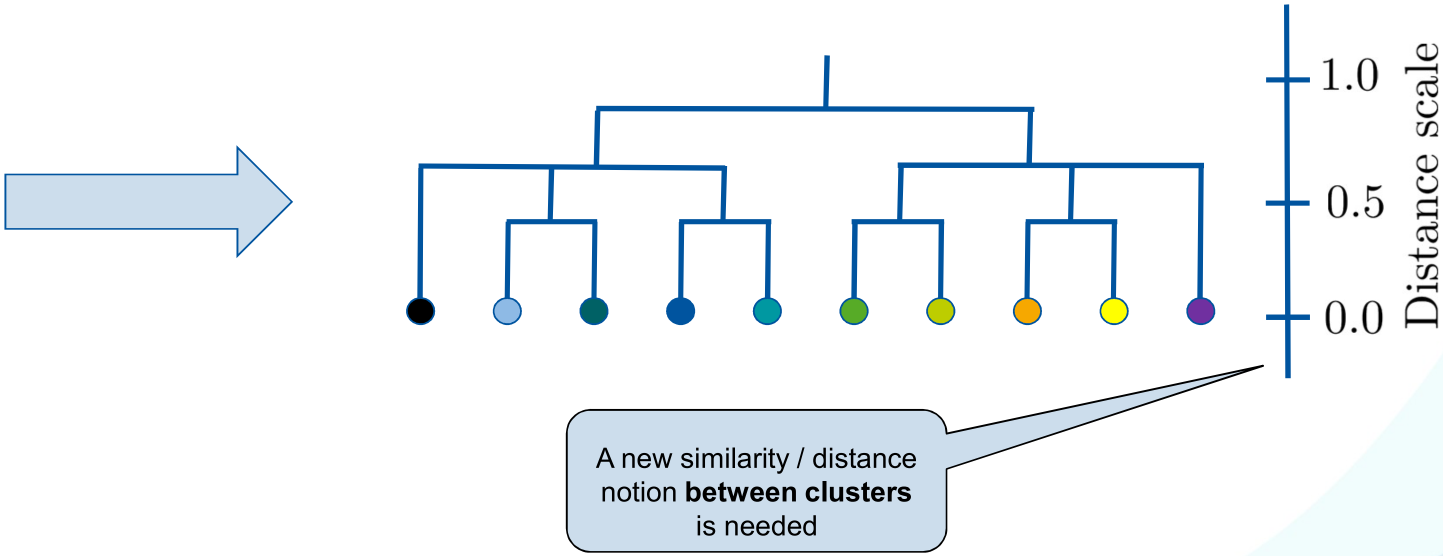
Hierarchical Clustering



Dendrogram

- Look for two clusters that are most similar and create a new cluster by merging them
- Value depicted when merged is the similarity / distance before merging





Linkage Measures

- The distance between clusters is otherwise known as linkage measure
- Four widely used linkage measures:

Distance between any two instances $\boldsymbol{x_n}$ and $\boldsymbol{x_m}$

Minimum distance: $\text{dist}_{\min}(\mathcal{C}_i, \mathcal{C}_j) = \min_{\mathbf{x_n} \in \mathcal{C}_i, \mathbf{x_m} \in \mathcal{C}_j} \{\|\mathbf{x_n} - \mathbf{x_m}\|\}$

Maximum distance: $\text{dist}_{\max}(\mathcal{C}_i, \mathcal{C}_j) = \max_{\mathbf{x_n} \in \mathcal{C}_i, \mathbf{x_m} \in \mathcal{C}_j} \{\|\mathbf{x_n} - \mathbf{x_m}\|\}$

$\boldsymbol{c_i}$ is the mean (centroid) of cluster $\boldsymbol{C_i}$

Mean distance: $\text{dist}_{\text{mean}}(\mathcal{C}_i, \mathcal{C}_j) = \|\mathbf{c_i} - \mathbf{c_j}\|$

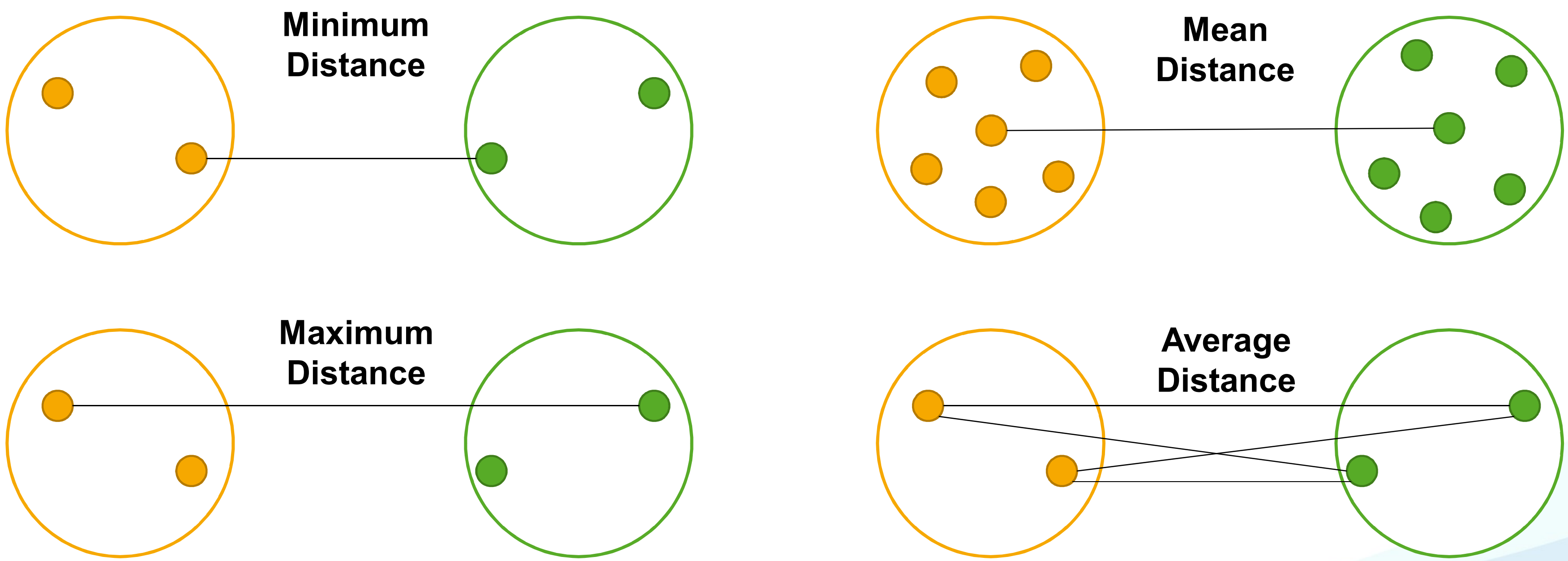
Average distance: $\text{dist}_{avg}(\mathcal{C}_i, \mathcal{C}_j) = \frac{1}{|C_i| \cdot |C_j|} \sum_{\mathbf{x_n} \in \mathcal{C}_i, \mathbf{x_m} \in \mathcal{C}_j} \|\mathbf{x_n} - \mathbf{x_m}\|$

Clusters $\boldsymbol{C_i}$ and $\boldsymbol{C_j}$

$\boldsymbol{|C_i|}$ is the number of instances in cluster $\boldsymbol{C_i}$

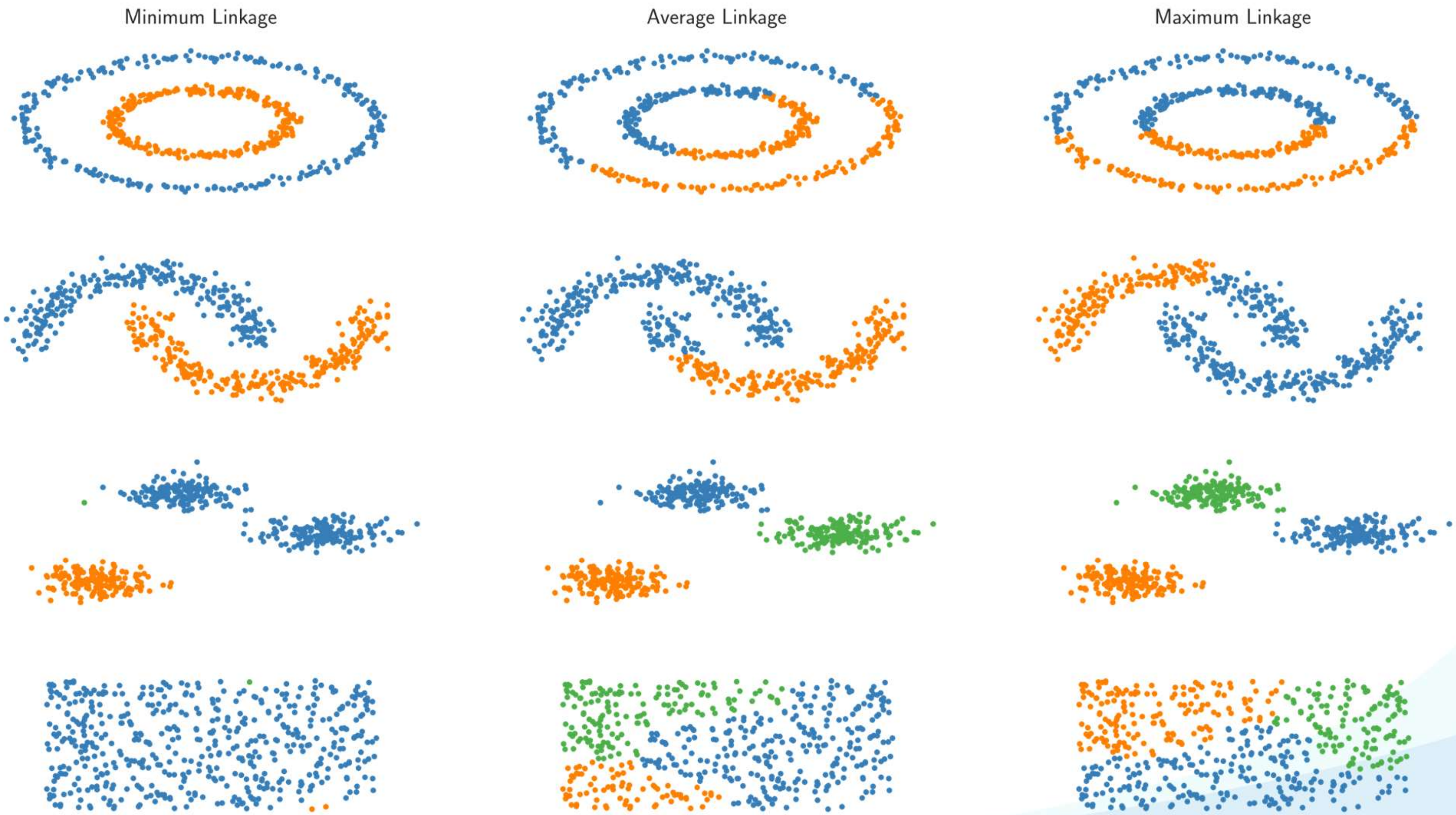
Linkage Measures

- The distance between clusters is otherwise known as linkage measure
- Four widely used linkage measures:



Linkage Measures

Different linkage measures may lead to different results



Algorithm

Simplistic version (many variants possible)

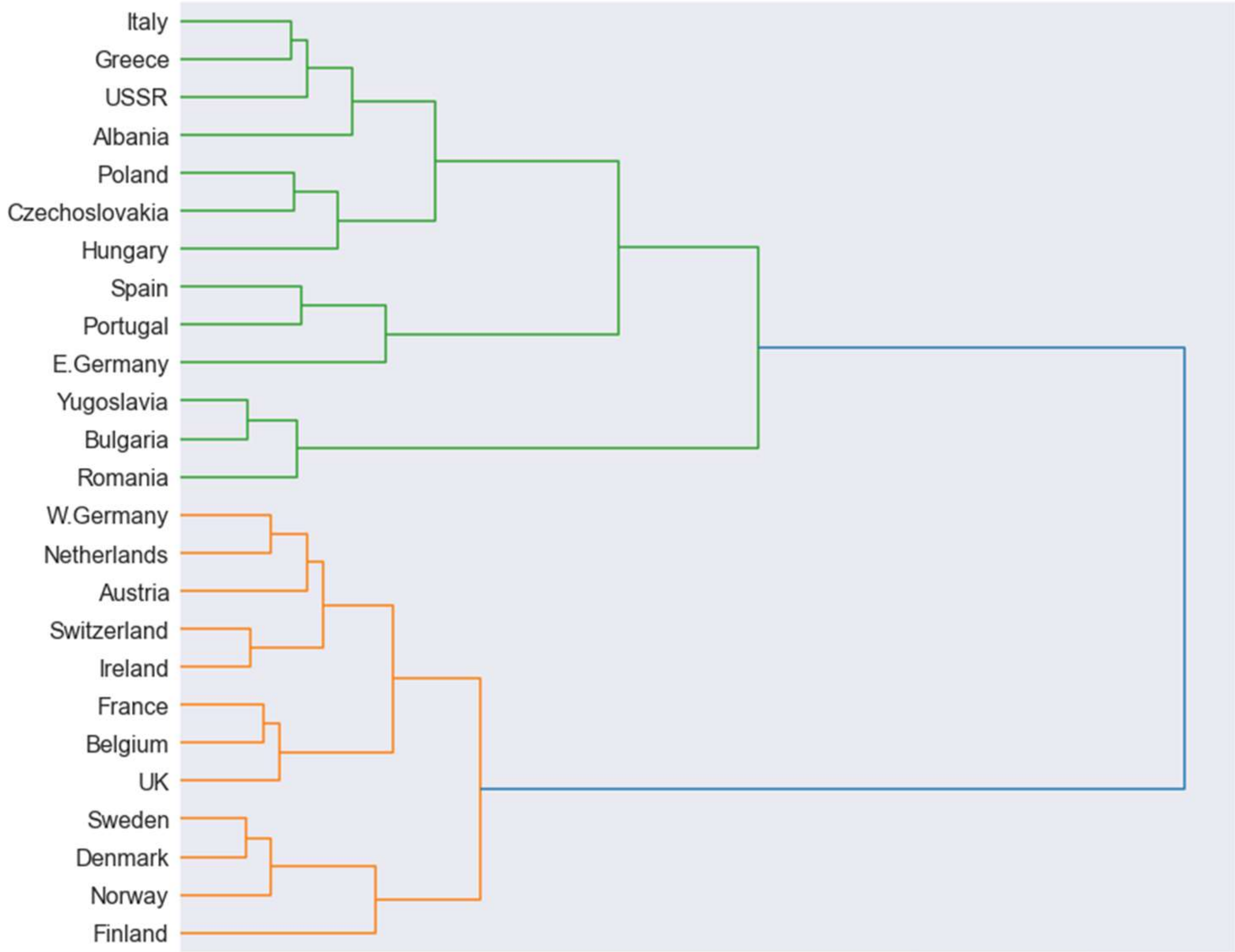
Agglomerative hierarchical clustering algorithm:

1. create a singleton cluster $\mathcal{C}_i$ for each instance $\mathbf{x_i}$
2. **repeat until one cluster is left**
   (a) compute the pairwise distance (using some linkage measure) between any two clusters $\mathcal{C}_i$ and $\mathcal{C}_j$
   (b) merge the two closest clusters
3. **return** dendrogram

Dendrogram – Example

Countries clustered by source of average protein consumption

Note that the agglomerative clustering procedure “discovers” geographic proximity!

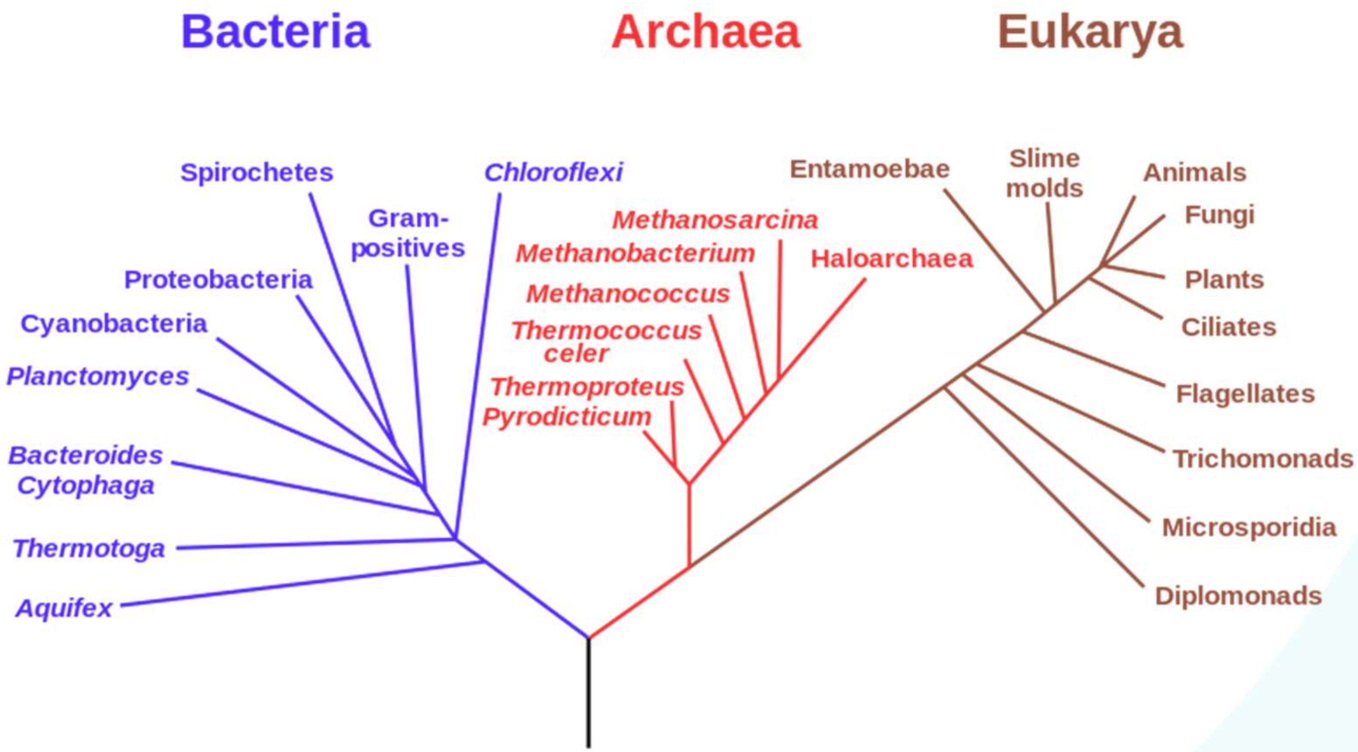


F. Karabiber

Dendrogram – Example

Phylogenetic trees:

Dendrograms obtained through clustering by genetic information (in this case, tRNA)

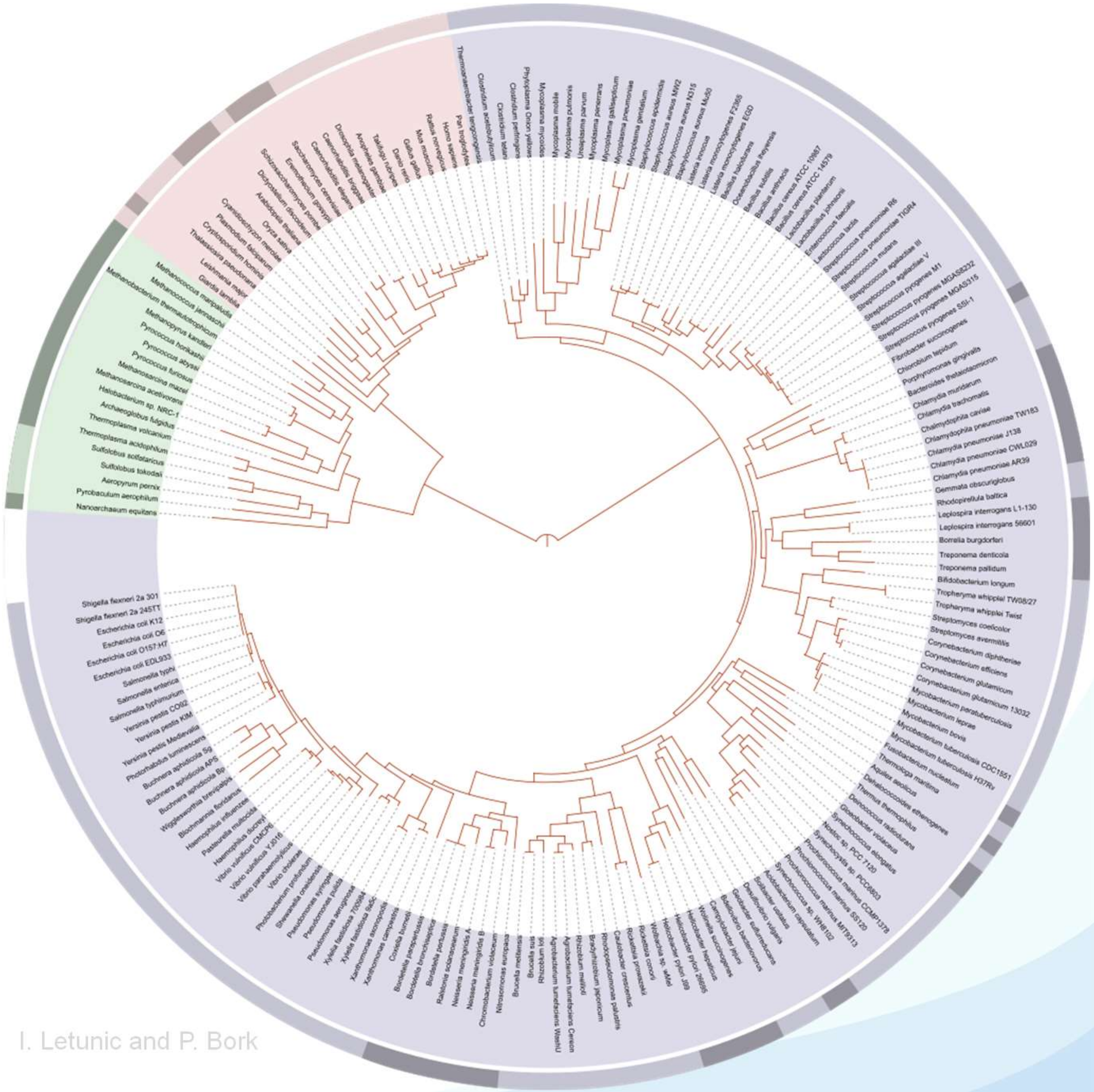


C.R. Woese et al.

Dendrogram – Example

Phylogenetic trees:

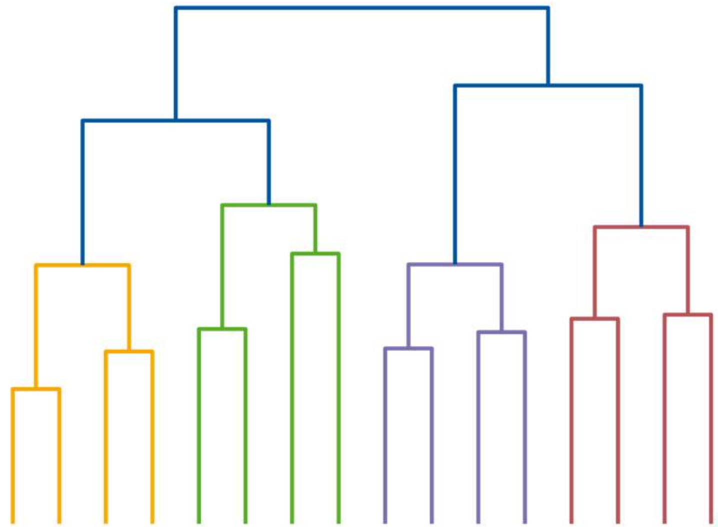
Dendrograms obtained through clustering by genetic information (in this case, full genome sequencing)



I. Letunic and P. Bork

Properties

- No a priori information / decision about the number of clusters is required
- Dendrogram allows analysts to “play” with abstraction level
- The algorithm cannot undo joins that turn out to be undesirable
- There is no approach to objectively minimize some well-defined errors



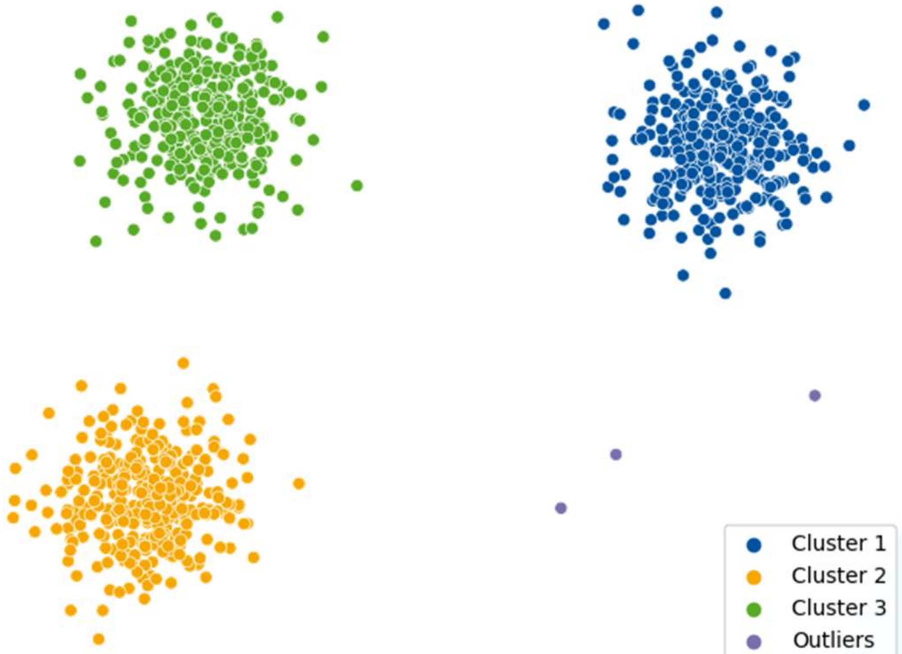
Density-Based Clustering

Density-Based Clustering

- Clusters are areas of **higher density**
- Used to find **clusters of any shape** (contrary to partitioning and hierarchical methods which tend to find spherical clusters)

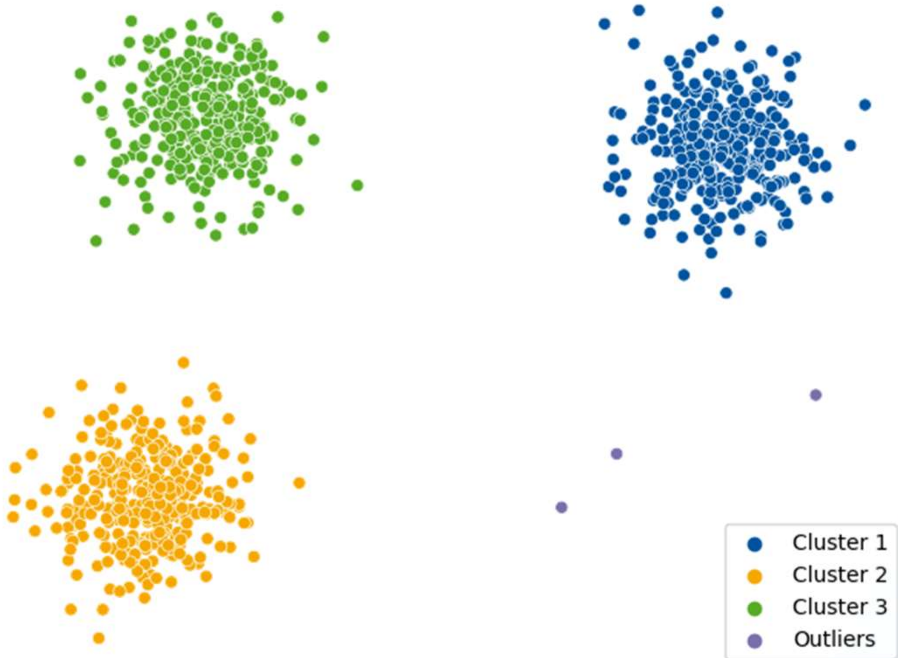
Density-Based Clustering

- Clusters are areas of **higher density**
- Used to find **clusters of any shape** (contrary to partitioning and hierarchical methods which tend to find spherical clusters)
- Instances in sparse areas are considered to be **outliers**
- Example – Density-Based Spatial Clustering of Applications with Noise (DBSCAN)



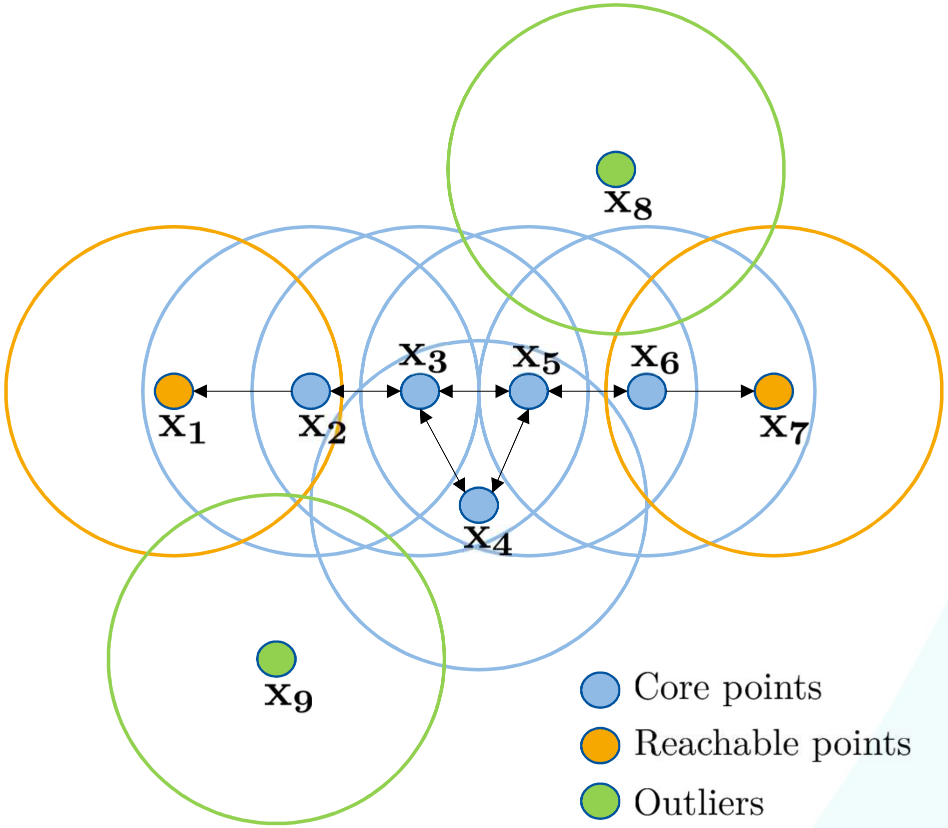
Density-Based Clustering

- Two instances $\mathbf{x_i}$ and $\mathbf{x_j}$ are **density-connected** if there is a core point $\mathbf{x_k}$ such that both $\mathbf{x_i}$ and $\mathbf{x_j}$ are reachable from $\mathbf{x_k}$
- Density-connectedness is **symmetric** (unlike reachability)
- A **cluster** satisfies the following two properties:
  - All instances within the cluster are **mutually density-connected**
  - Any two density-connected core points are part of the cluster



DBSCAN

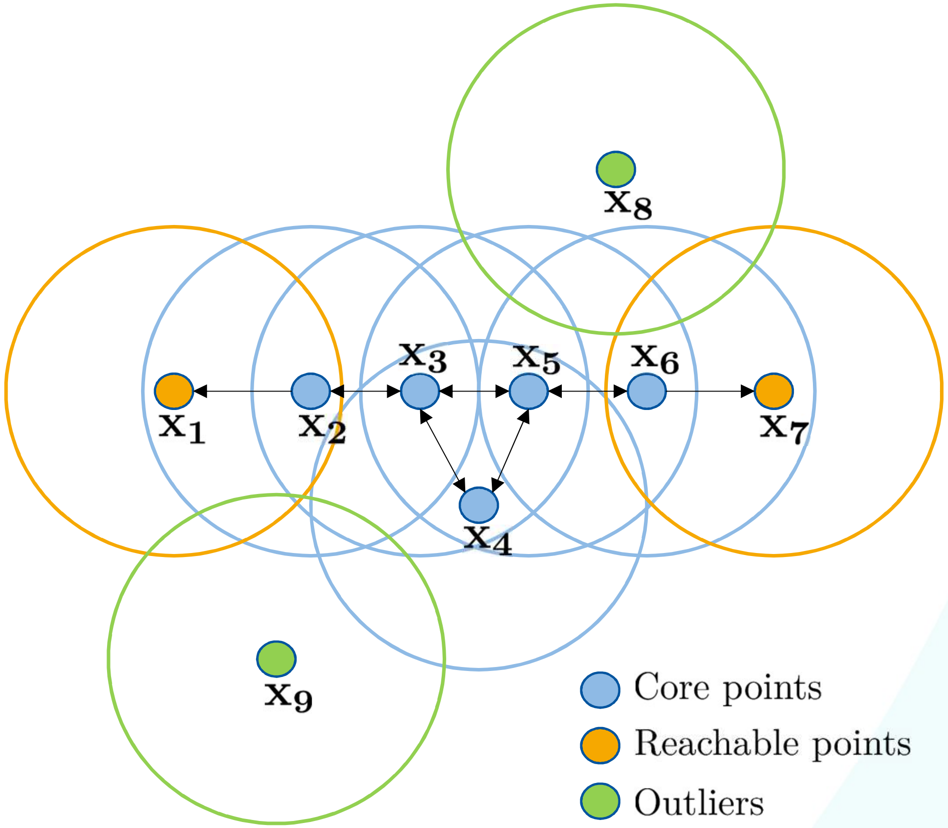
- Two parameters:
  - ϵ (fixed neighborhood size)
  - $MinPts$ (density threshold for dense regions)
- ϵ is the maximum radius of the neighborhood from $\mathbf{x}_i$
- Instance $\mathbf{x}_i$ is a core point if at least $MinPts$ are within distance ϵ (including $\mathbf{x}_i$)



ϵ is indicated by circles and $MinPts = 3$

DBSCAN - Example

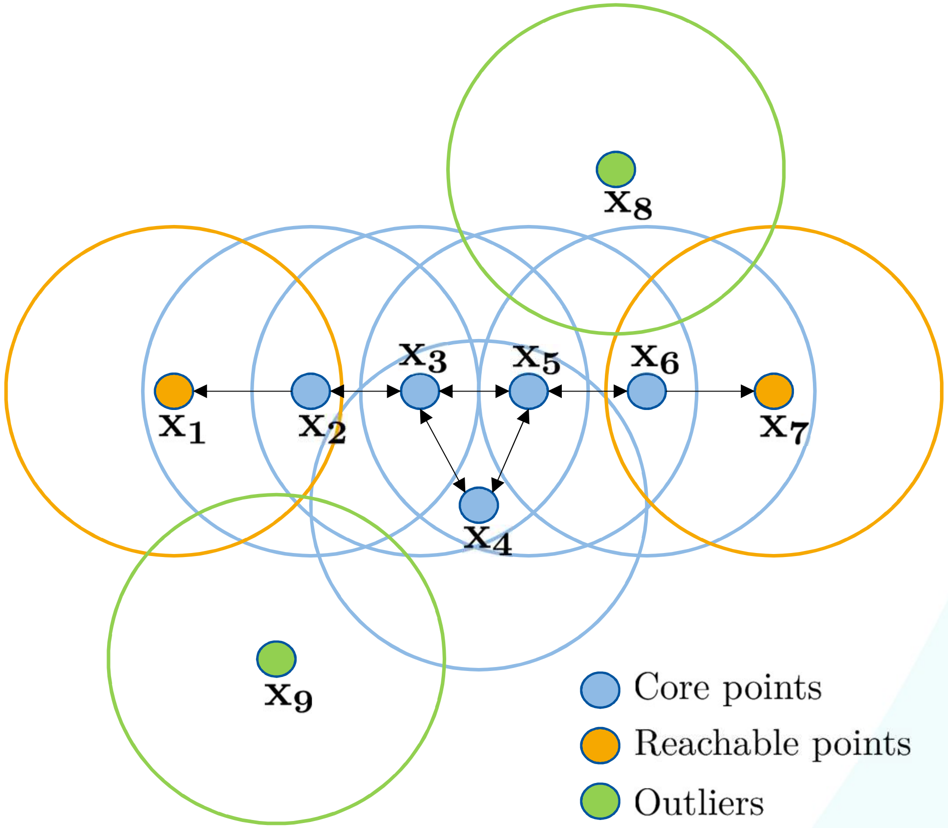
- An instance $\mathbf{x_j}$ is **directly reachable** from $\mathbf{x_i}$ if $\mathbf{x_j}$ is within distance ϵ from $\mathbf{x_i}$ and $\mathbf{x_i}$ is a **core point**



ϵ is indicated by circles and $MinPts = 3$

DBSCAN - Example

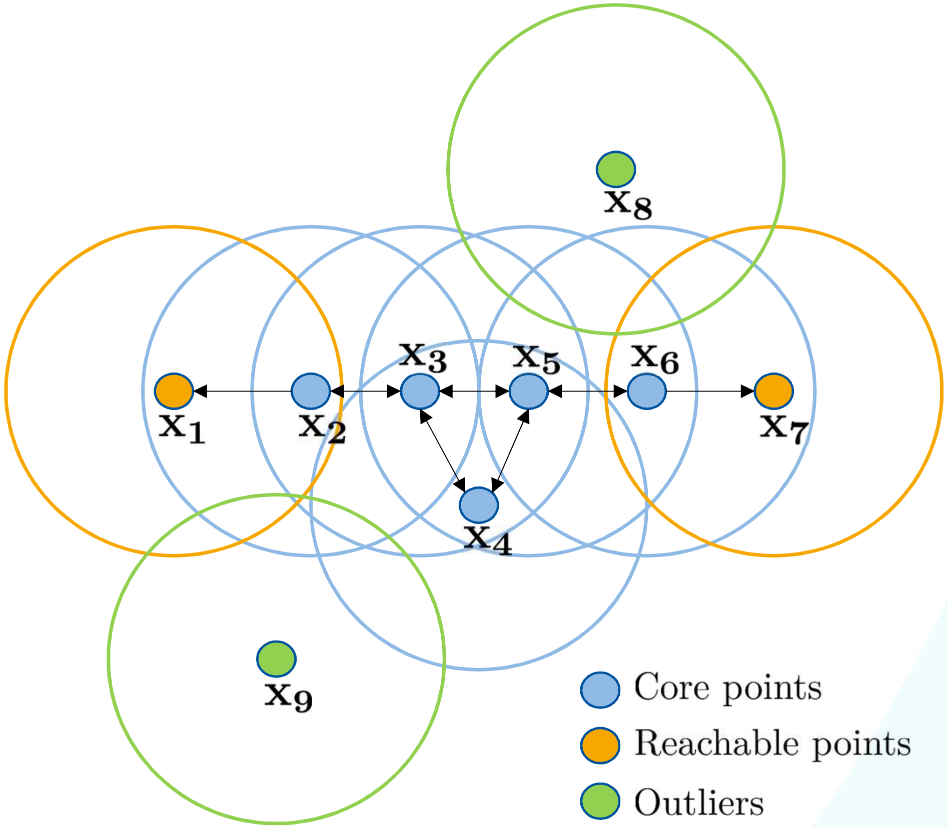
- An instance $\mathbf{x_j}$ is **directly reachable** from $\mathbf{x_i}$ if $\mathbf{x_j}$ is within distance ϵ from $\mathbf{x_i}$ and $\mathbf{x_i}$ is a **core point**
- An instance $\mathbf{x_j}$ is **reachable** from $\mathbf{x_i}$ if there is a path $\langle \mathbf{y_1}, \mathbf{y_2}, \ldots, \mathbf{y_K} \rangle$ with $\mathbf{y_1} = \mathbf{x_i}$ and $\mathbf{y_K} = \mathbf{x_j}$ where each $\mathbf{y_k}$ is directly reachable from $\mathbf{y_{k-1}}$
- All the points on the path must be **core points**, except for $\mathbf{x_j}$, i.e., $\mathbf{y_1}, \mathbf{y_2}, \ldots, \mathbf{y_{n-1}}$ are core points



ϵ is indicated by circles and $MinPts = 3$

DBSCAN - Example

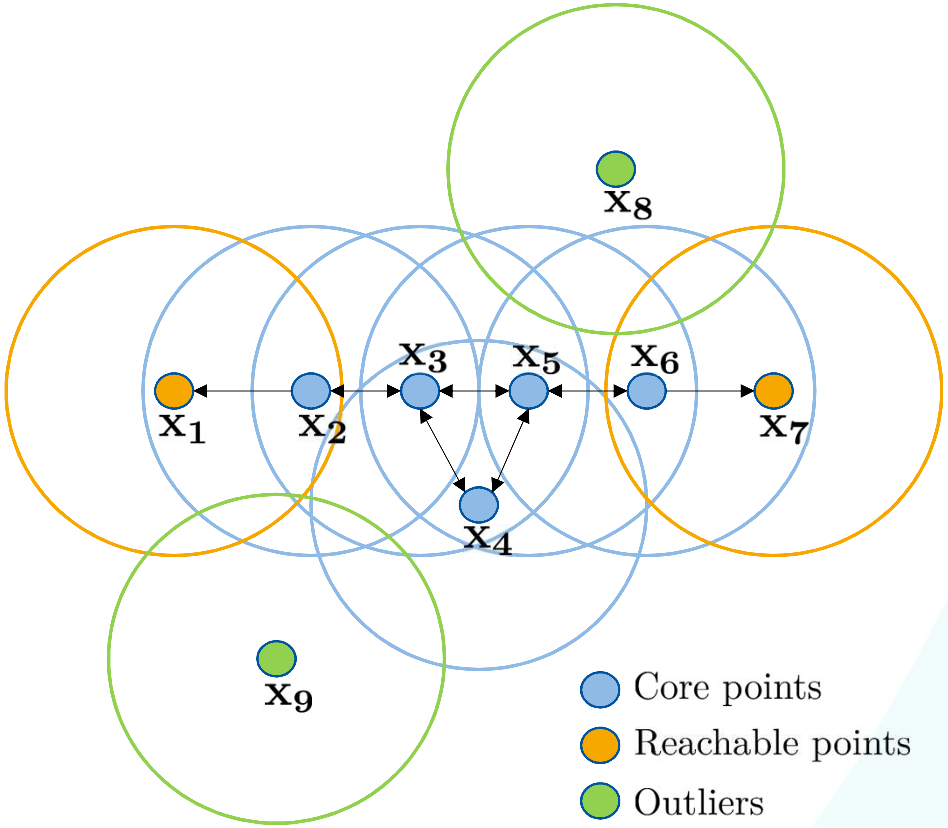
- An instance $\mathbf{x_j}$ is **directly reachable** from $\mathbf{x_i}$ if $\mathbf{x_j}$ is within distance ϵ from $\mathbf{x_i}$ and $\mathbf{x_i}$ is a **core point**
- An instance $\mathbf{x_j}$ is **reachable** from $\mathbf{x_i}$ if there is a path $\langle \mathbf{y_1}, \mathbf{y_2}, \dots, \mathbf{y_K} \rangle$ with $\mathbf{y_1} = \mathbf{x_i}$ and $\mathbf{y_K} = \mathbf{x_j}$ where each $\mathbf{y_k}$ is directly reachable from $\mathbf{y_{k-1}}$
- All the points on the path must be **core points**, except for $\mathbf{x_j}$, i.e., $\mathbf{y_1}, \mathbf{y_2}, \dots, \mathbf{y_{n-1}}$ are core points
- All points **not reachable** from any other point are **outliers**



ϵ is indicated by circles and $MinPts = 3$

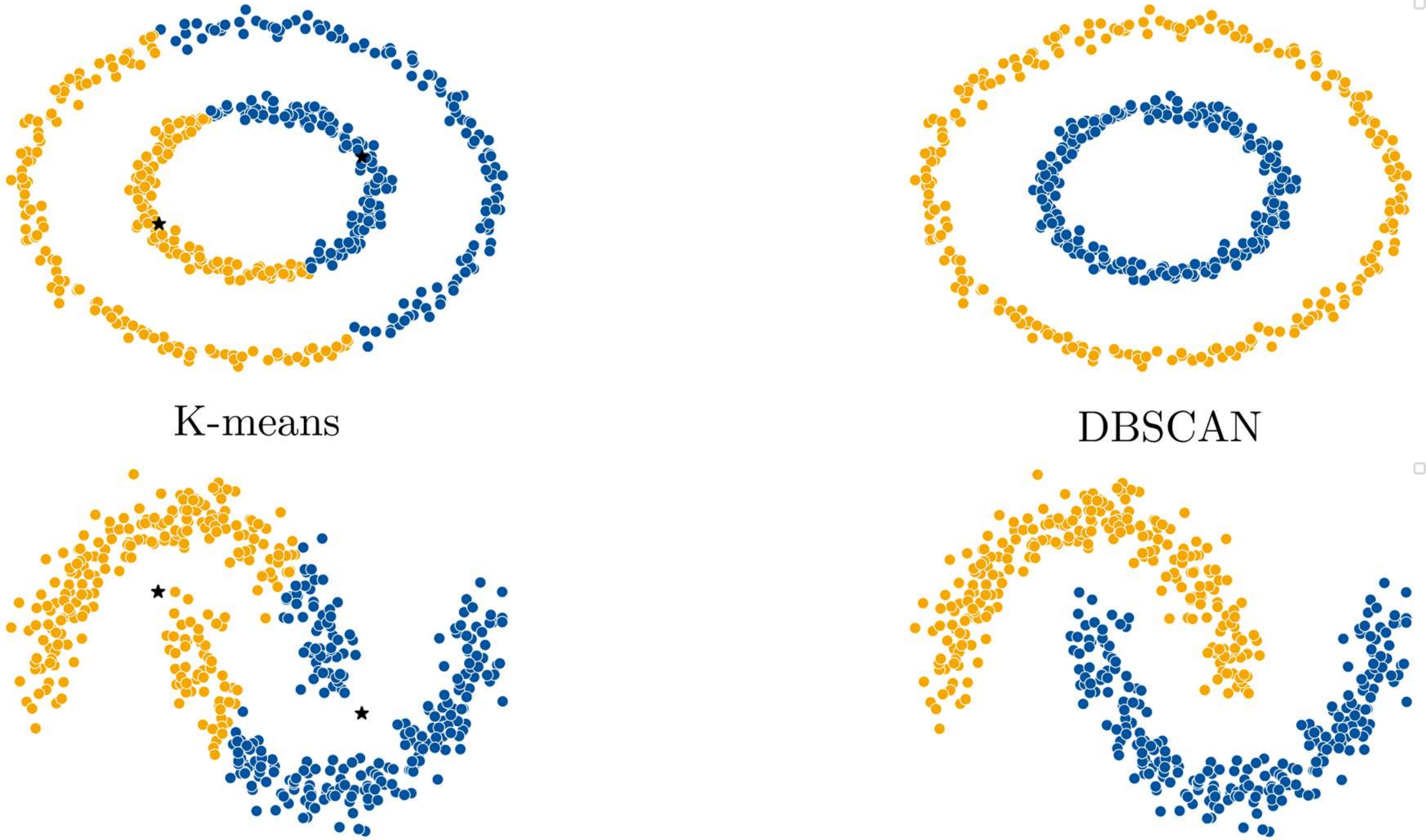
DBSCAN - Approach

- The approach iteratively selects a **core point $\mathbf{x}_i$ not yet part of a cluster** and creates a cluster for it
- The cluster is **incrementally extended** by adding all neighboring points of core points in the cluster
- If a cluster cannot be extended anymore, the next unvisited core point is considered
- The approach is **not entirely deterministic**: Points reachable from more than one cluster are assigned based on the processing order



ϵ is indicated by circles and $MinPts = 3$

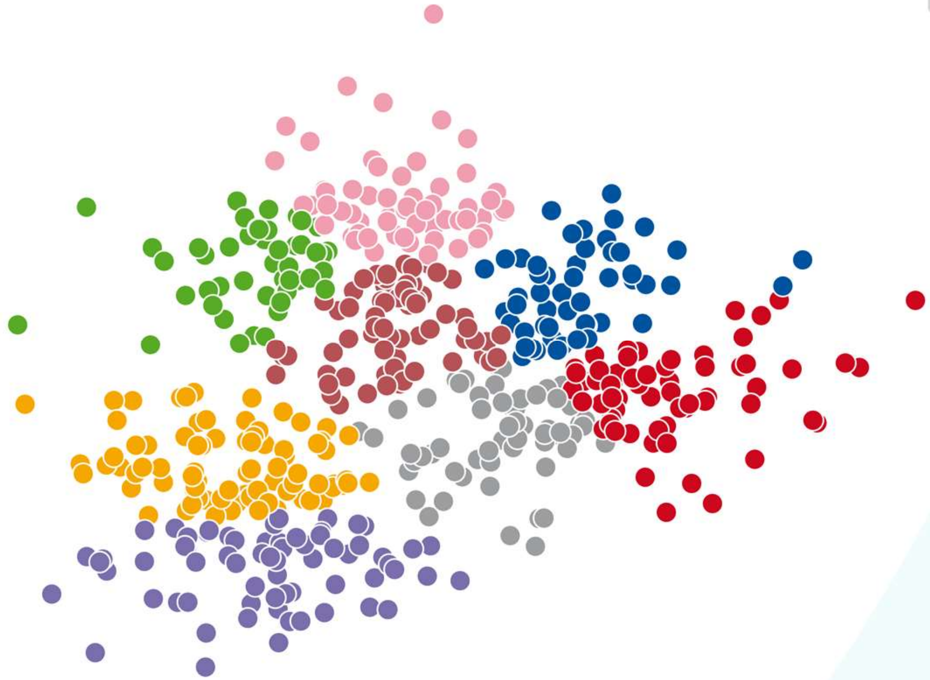
DBSCAN - Graphical Examples



Conclusion

General Problem – Cluster Interpretation

- Cluster quality may be good, but this does not imply that the clusters reveal new insights
- Describe clusters in terms of their features (e.g., compare centroids)
- Use simple visualization techniques like boxplots to compare clusters



Takeaways

- Clustering: grouping together unlabeled instances
- Useful for explorative analysis, and when choosing a label does not make sense
- However, results are often hard to validate!
- Various approaches:
  - Based on (spatial) distance
  - By agglomeration
  - By density
- And many more!

- Next up: Frequent Itemsets