Unveiling the Transformative Power of Unsupervised machine learning through Clustering

Vishnu S. Pendyala, Ph.D.
San Jose State University

To cite this presentation: Pendyala, V.S. (2025) "Unveiling the Transformative Power of Unsupervised machine learning through Clustering". IEEE Computer Society, Kitchener-Waterloo Chapter Technical Talk, March 31, 2025

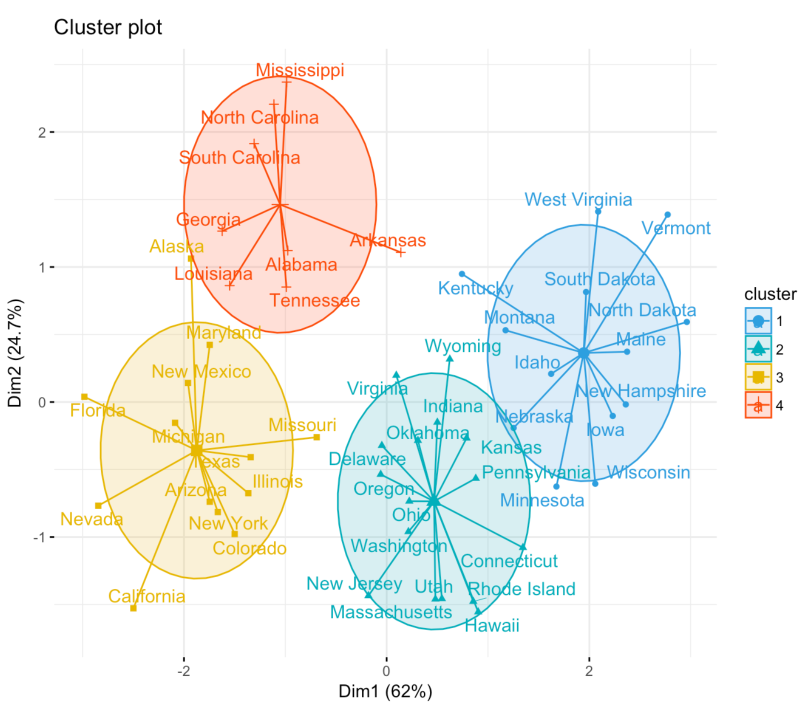
How do Streaming services know that these movies can be grouped together?

Features!

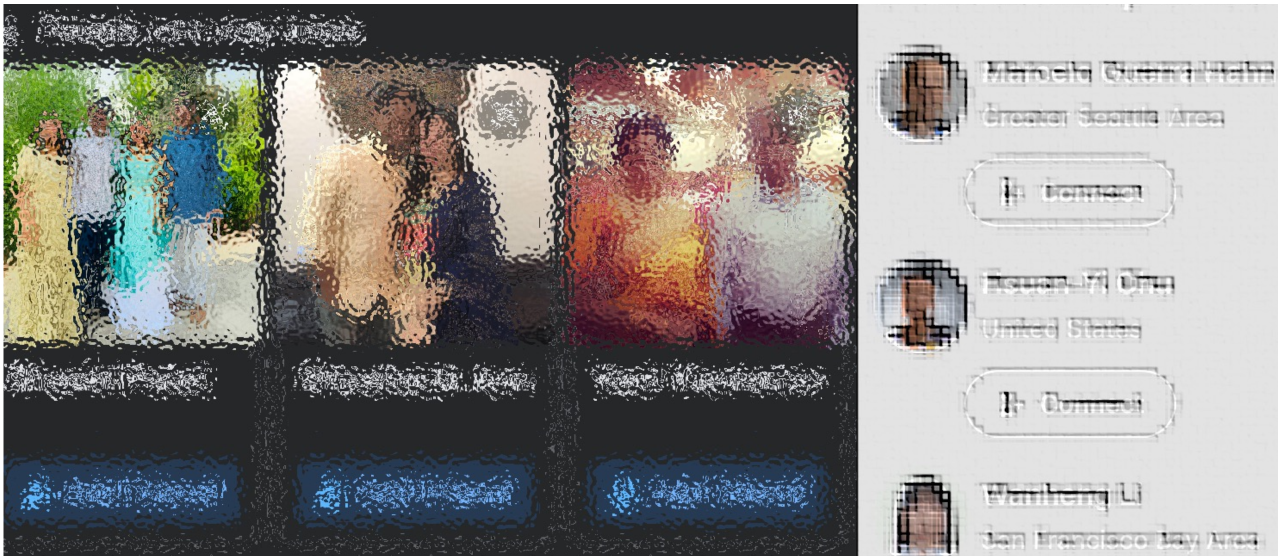
Features are expressed as vectors

Feature	Value
Genre (Action)	8
Genre (Drama)	7
Genre (War)	6
Historical Accuracy	7
Heroism Level	9
Number of Battles	5
Emotional Depth	8
IMDB User Rating	8.4

Feature vector x_i =[8, 7, 6, 7, 9, 5, 8, 8.4]

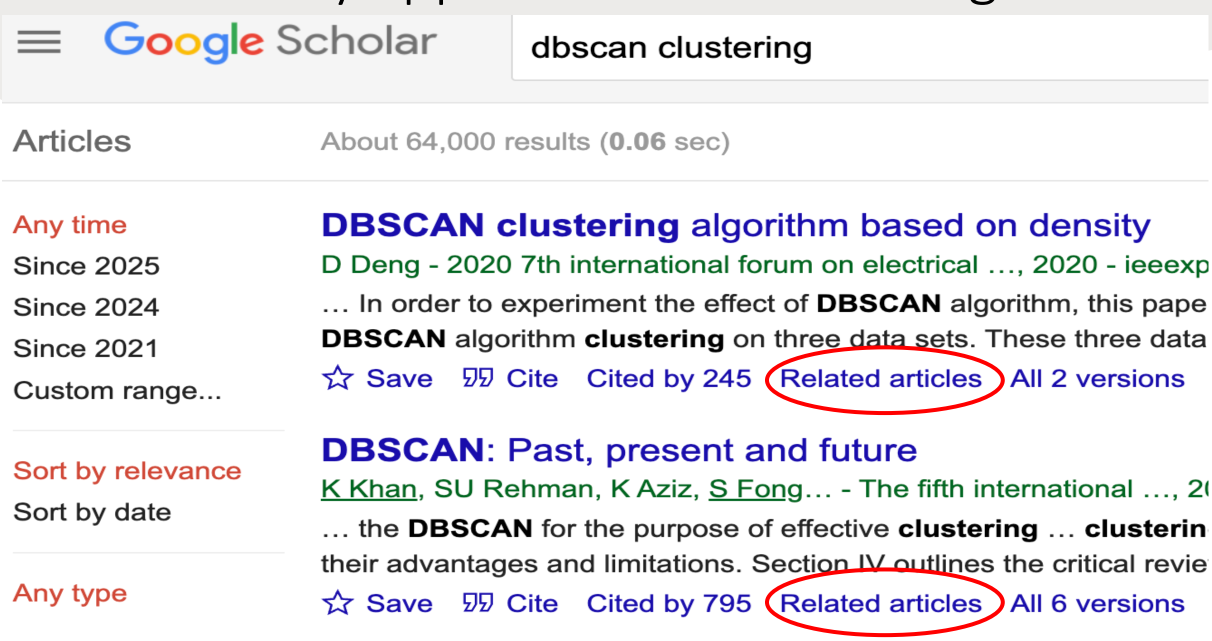


- Clustering relies on distances in the feature space
- May not correspond to physical distance
- Distance is a measure for similarity
- Smaller distance => better similarity
- Inter-cluster distances must be maximized
- Intra-cluster distances must be minimized
- No labels (unsupervised)
- Labels => classification



Some applications of Clustering: **Social Media**

Many applications of clustering



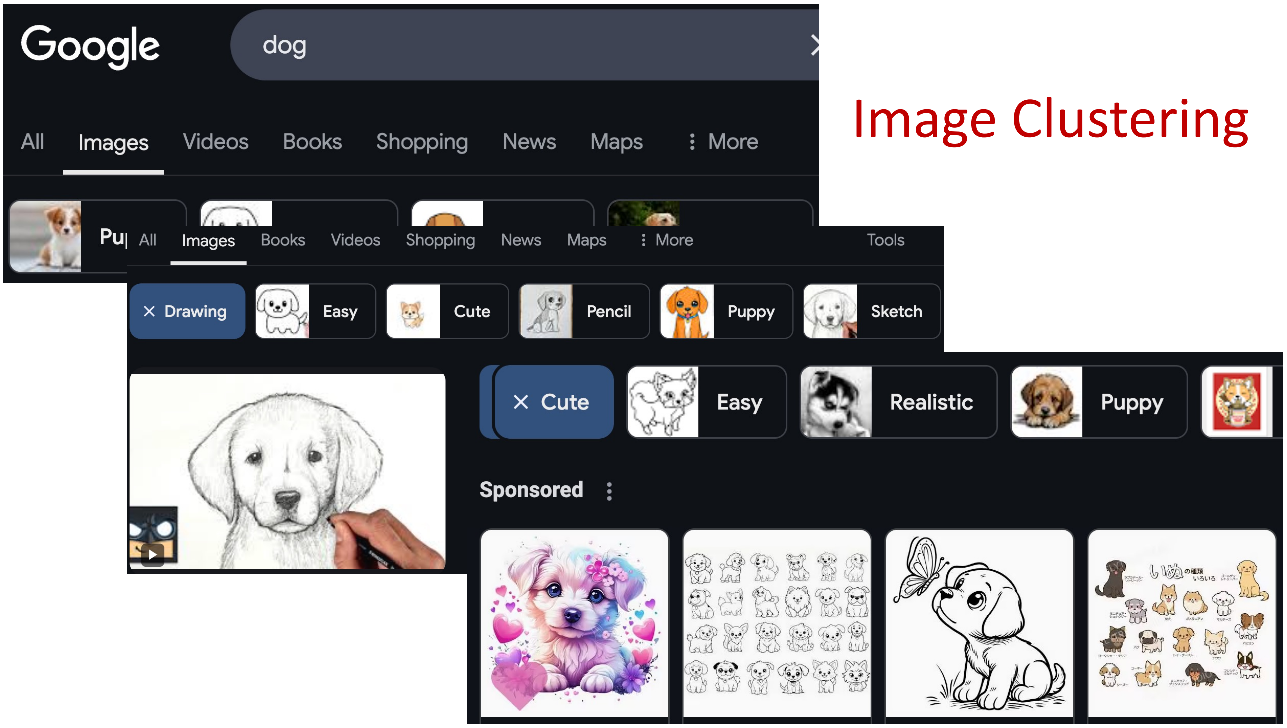


Image Clustering

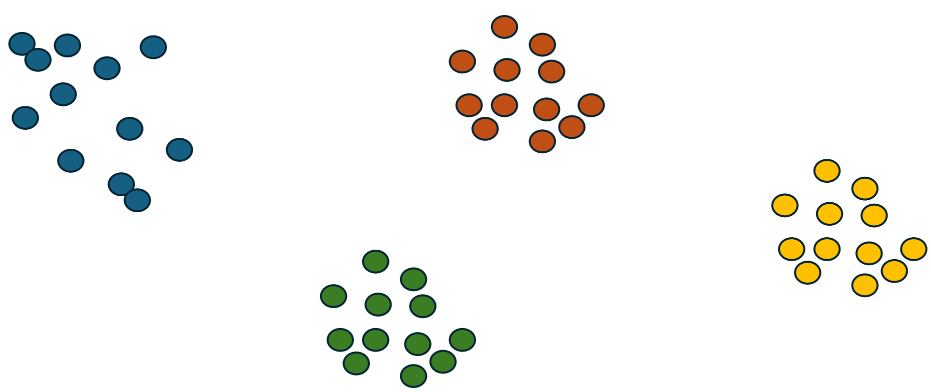
Classes vs Clusters

- Classes are defined and data labeled manually (Supervised)
- Clusters are deduced automatically, no labels (Unsupervised)

Hyperparameters: What can you choose for clustering data?

- Similarity metric: Cosine, Euclidean, Manhattan, Geodesic, …
- Type of clustering: partitioning (non-overlapping subsets), Hierarchical Clustering (tree-like), Density-Based , Fuzzy Clustering (points to belong to multiple clusters with varying degrees of membership), …
- Clustering algorithm: K-Means, Agglomerative Clustering, Divisive Clustering, DBSCAN,…
- Number of clusters – some algorithms like K-Means need this

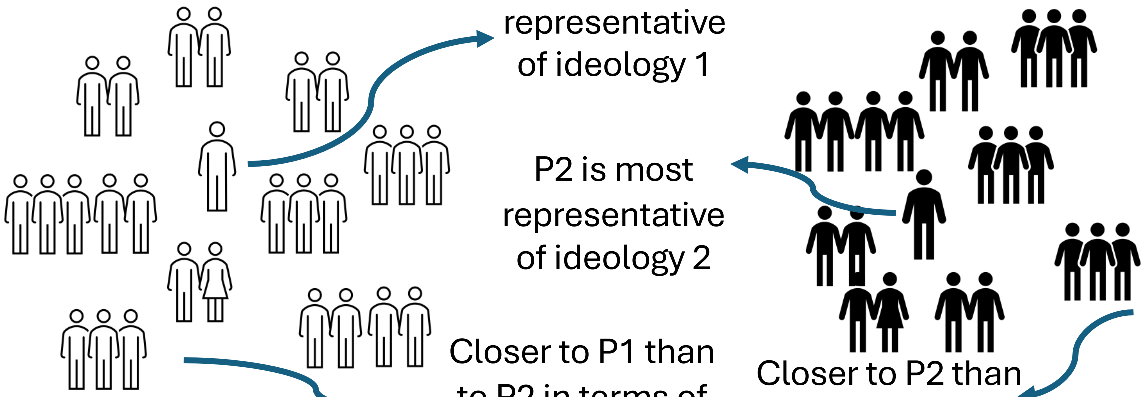
How can we detect and form these clusters of data points programmatically?



How are clusters formed in real life?

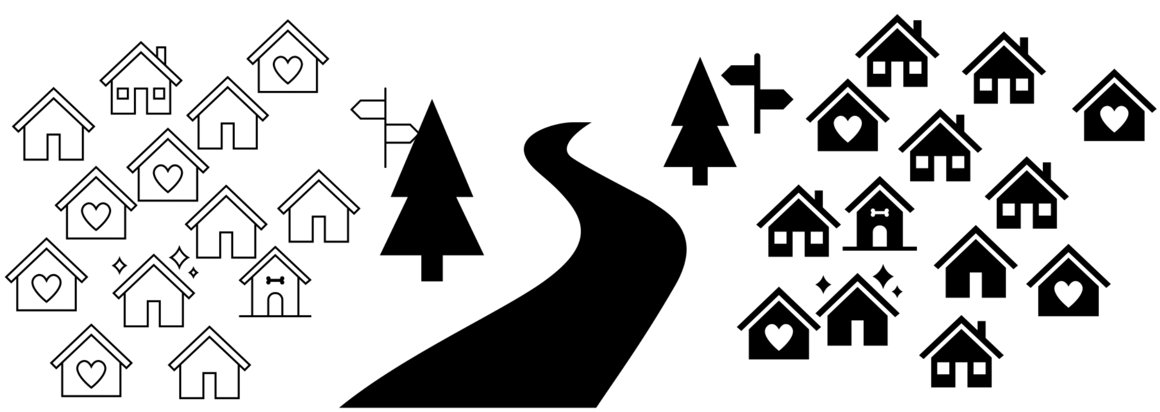
Political Parties

K-means



How are clusters formed in real life?

Neighborhoods

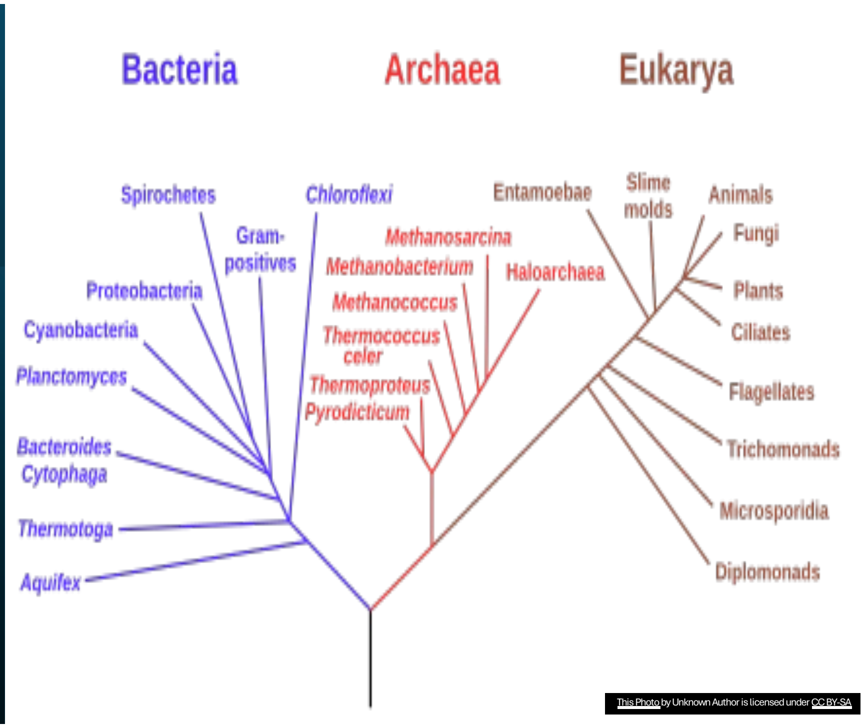


DBSCAN

Hierarchical

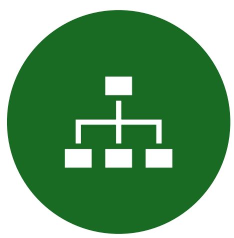
How are clusters formed in real life?

Biological Species

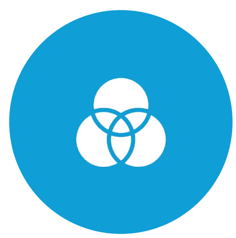


Clustering algorithms for today

K-MEANS

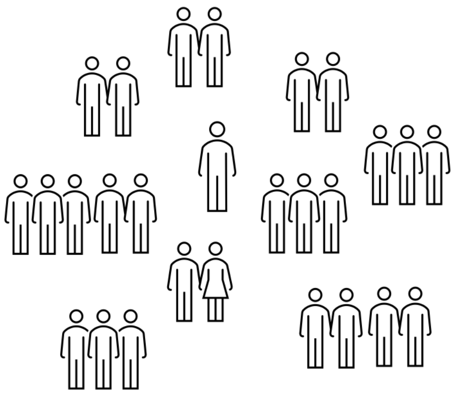


HIERARCHICAL CLUSTERING

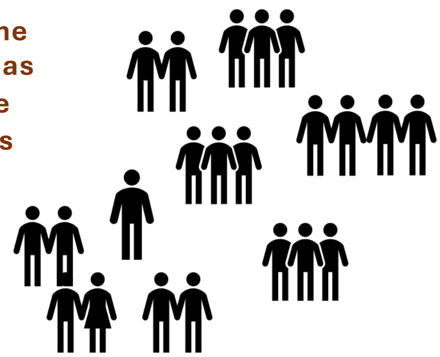


DENSITY-BASED SPATIAL CLUSTERING OF APPLICATIONS WITH NOISE (DBSCAN)

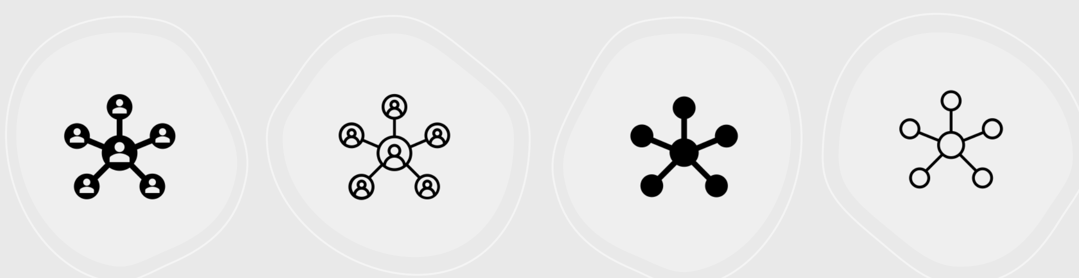
The Goal of Clustering



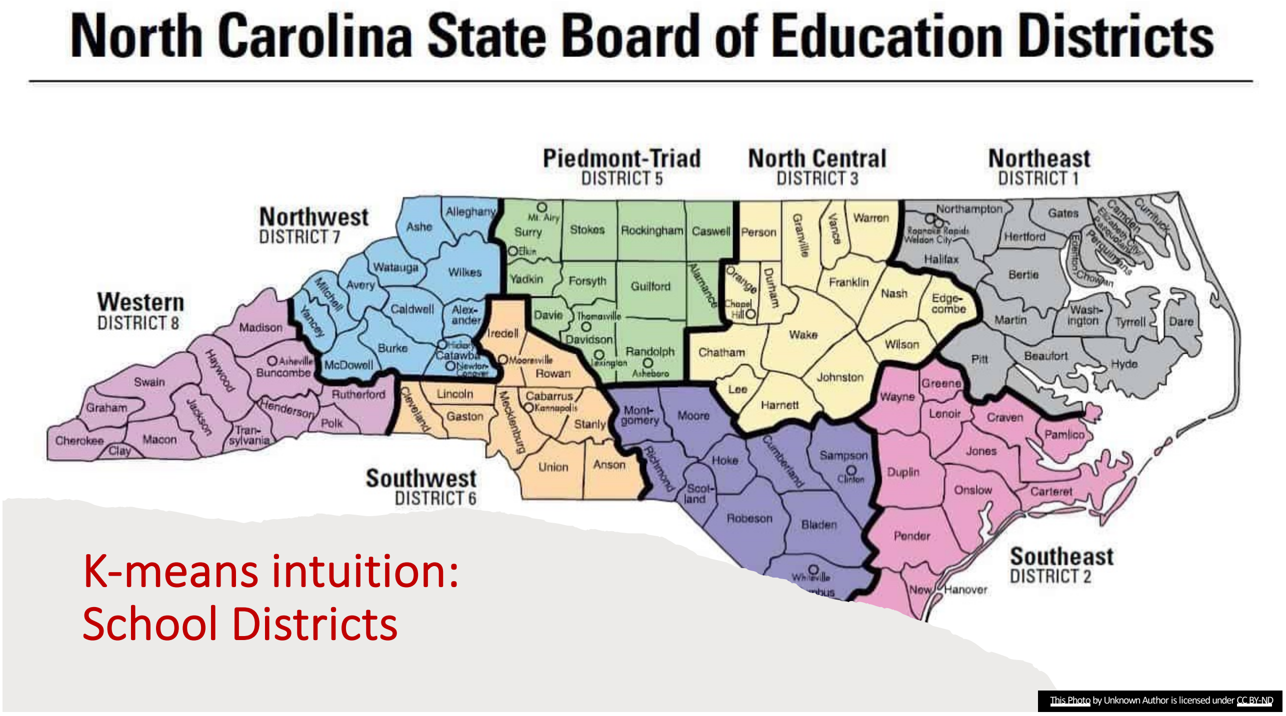
Entities across the clusters must be as **dissimilar** in the feature space as possible

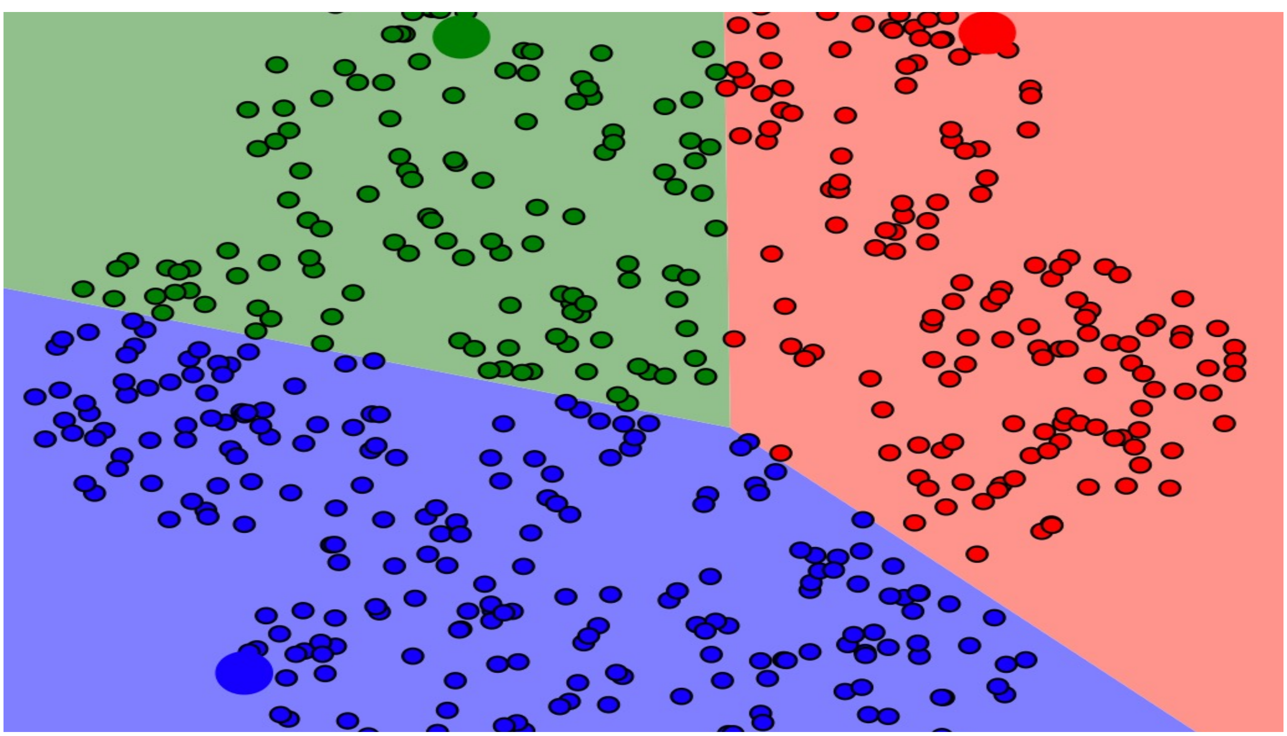


Entities within the clusters must be as **similar** in the feature space as possible



K-Means Clustering


North Carolina State Board of Education Districts
Piedmont-Triad
DISTRICT 5
North Central
DISTRICT 3
Northeast
DISTRICT 1
Northwest
DISTRICT 7
Western
DISTRICT 8
Southwest
DISTRICT 6
Southeast
DISTRICT 2
K-means intuition:
School Districts
This Photo by Unknown Author is licensed under CC BY-ND

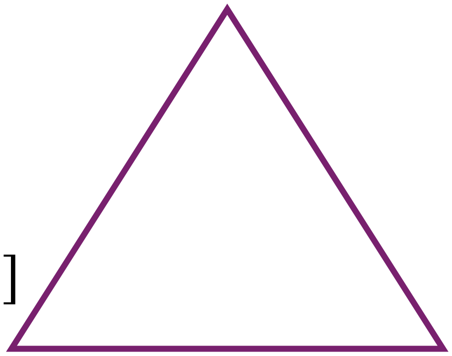


Central to K-means: Centroid

$$\mu = (\frac{x_1 + x_2 + x_3}{3}, \frac{y_1 + y_2 + y_3}{3})$$

$[x_1, y_1]$

$$\mu = \frac{1}{3}(\sum_{i=1}^{3} x_i , \sum_{i=1}^{3} y_i)$$

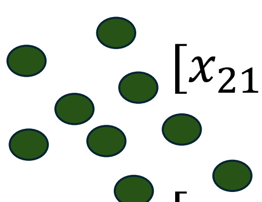


$[x_2, y_2]$ $[x_3, y_3]$

Centroid of a cluster

$[x_{11}, x_{12}, \dots, x_{1n}]$

$$\mu = \frac{1}{m}\left(\sum_{i=1}^{m} x_{i1} , \sum_{i=1}^{m} x_{i2} , \dots , \sum_{i=1}^{m} x_{in}\right)$$



$[x_{21}, x_{22}, \dots, x_{2n}]$

$$\vec{\boldsymbol{\mu}}_{\boldsymbol{c}} = \frac{\mathbf{1}}{|\boldsymbol{c}|}(\sum_{\boldsymbol{i}=\mathbf{1}}^{|\boldsymbol{c}|} \vec{\boldsymbol{x}}$$

$[x_{k1}, x_{k2}, \dots, x_{kn}]$

$[x_{m1}, x_{m2}, \dots, x_{mn}]$

$\boldsymbol{m\ data\ points}$

Clustering Hyperparameter: Similarity Metric

Metric	Formula	Properties	Best Used For	Limitations
Euclidean Distance	$\sqrt{\sum_{i=1}^{n}(x_i - y_i)^2}$	- Intuitive - Preserves original space	- Continuous data - Low-dimensional spaces	- Sensitive to outliers - Struggles with high dimensions
Manhattan Distance	$\sum_{i=1}^{n}\lvert x_i - y_i\rvert$	- Less sensitive to outliers - Computationally efficient	- Grid-like path problems	- Less intuitive - May not capture diagonal relationships
Minkowski Distance	$\left(\sum_{i=1}^{n}\lvert x_i - y_i\rvert^p\right)^{1/p}$	- Generalizes Euclidean (p=2) and Manhattan (p=1) - Flexible parameter p	- When optimal distance metric is unknown - Tuning to specific datasets	- Parameter selection can be challenging - Computationally intensive for non-integer p

Distance metrics (continued)

Metric	Formula	Properties	Best Used For	Limitations
Cosine Similarity	$\frac{\sum_{i=1}^{n} x_i\ y_i}{\sqrt{\sum_{i=1}^{n} x_i^2}\sqrt{\sum_{i=1}^{n} y_i^2}}$	- Measures angle, not magnitude - Bounded between -1 and 1	- Text analysis - Recommender systems - When direction matters more than magnitude	- Not a true metric (triangle inequality) - Undefined for zero vectors
Mahalanobis Distance	$\sqrt{(x-y)^T\Sigma^{-1}(x-y)}$	- Accounts for correlations - Scale-invariant	- Correlated features - Outlier detection - Classification tasks	- Requires covariance matrix estimation - Computationally expensive

Distance metrics (continued)

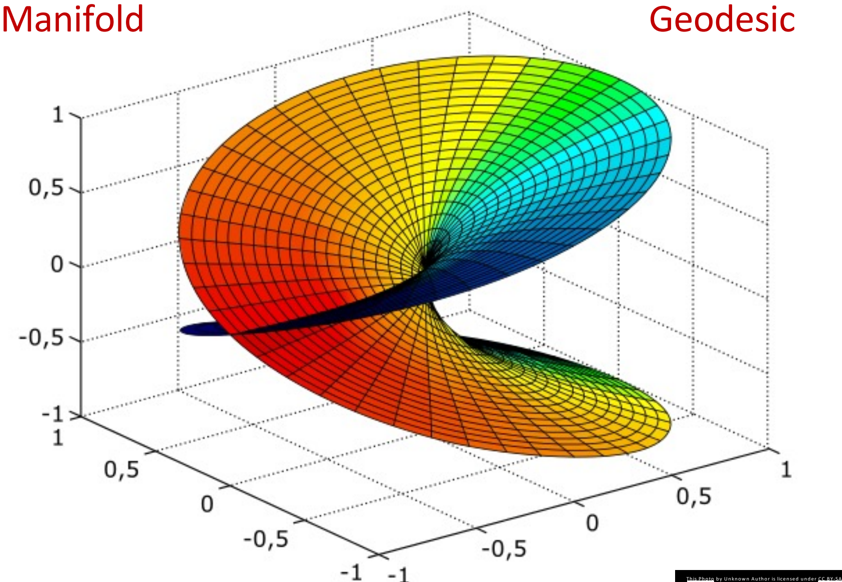
Metric	Formula	Properties	Best Used For	Limitations
Hamming Distance	Count of positions where vectors differ	- Simple to compute - Natural for categorical data	- Binary / categorical features - Error detection	- Limited to same-length sequences - No concept of magnitude
Jaccard Distance	$1 - \frac{\|A \cap B\|}{\|A \cup B\|}$	- Ratio-based - Bounded between 0 and 1	- Set-based problems - Document similarity	- Ignores frequency - Sensitive to small sets
Chebyshev Distance	$\max_i(\|x_i - y_i\|)$	- Determined by maximum difference - Fast to compute	- Warehouse/path logistics - When worst-case difference matters	- Ignores differences in other dimensions - Sensitive to outliers in single dimension

What is the problem with Euclidean distance?

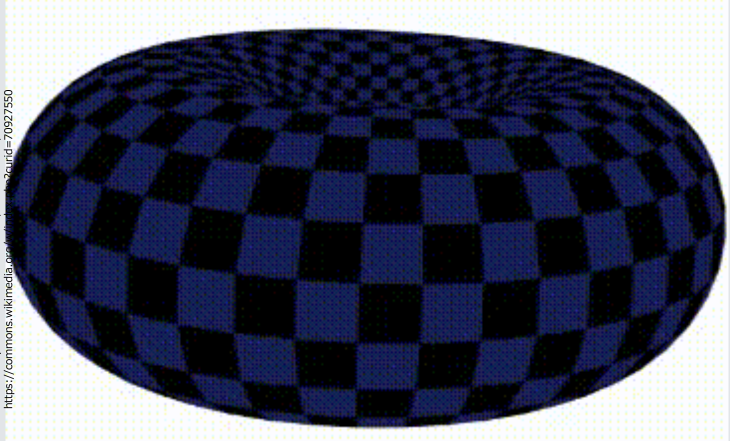
It assumes that the points are all on a hyperplane

Manifold

Geodesic



Geodesic – the shortest distance between two points on a manifold surface, honoring the shape



The K-Means Algorithm

- Most popular unsupervised machine learning for partitioning data into K disjoint clusters based on features
- Goal: Minimize within-cluster variance (dissimilarity, measured by distance or sum of squares)

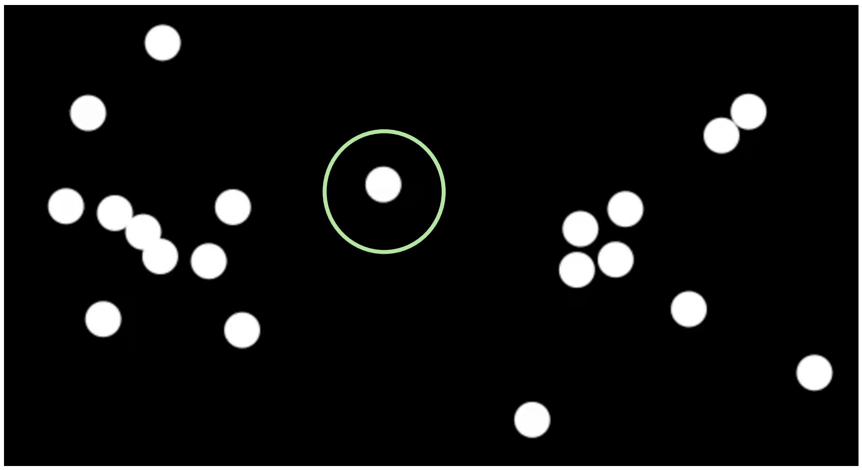
$$Z = \sum_{j=1}^{K} \sum_{n \in C_j} \left| x_n - \mu_j \right|^2$$

- x_n is a vector representing the n^{th} data point, μ_j is the centroid of the data points in the cluster C_j, and $\left| x_n - \mu_j \right|^2$ is the Euclidean distance between them.
- Widely used in segmentation problems, quantization, and hidden pattern (structure) recognition

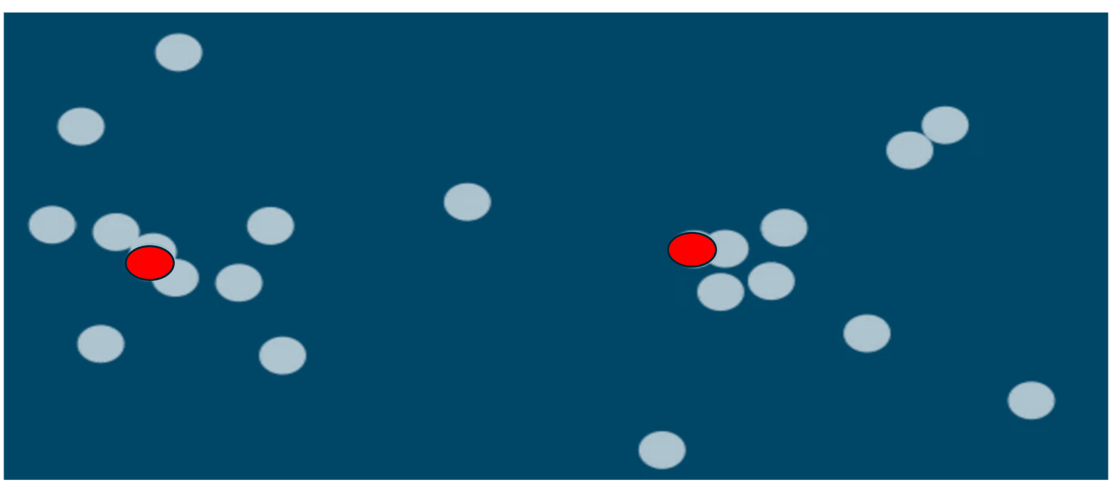
Why is μ_j the centroid?

- Intra-cluster distance $L = \Sigma(x_n - \mu_j)^2$
- $\frac{\partial L}{\partial \mu} = 2\Sigma(x_n - \mu_j) = 0$
- => $\mu_j = \frac{\Sigma x_n}{P}$, which is the formula for the centroid, where P is the number of points in the cluster
- Proves centroid (or the mean) is the most representative point in the cluster

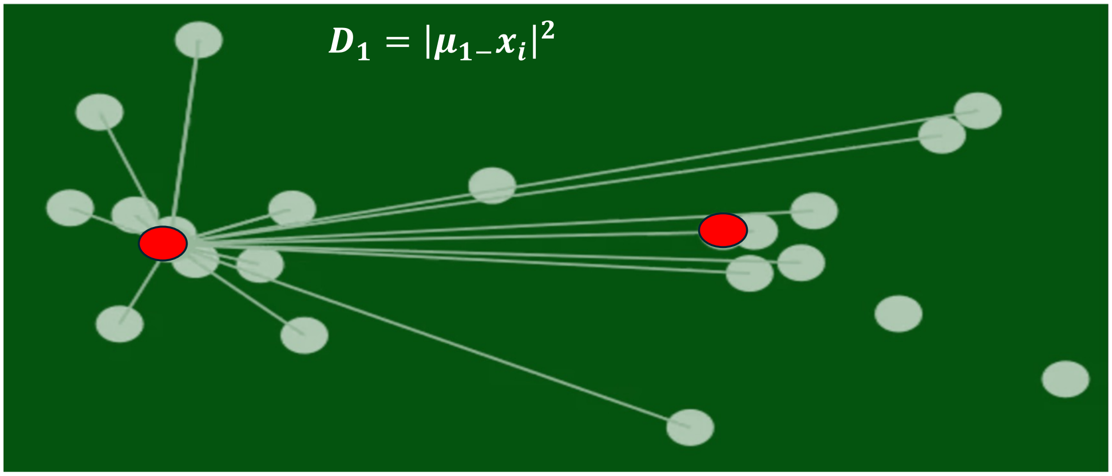
Initial Set of Points



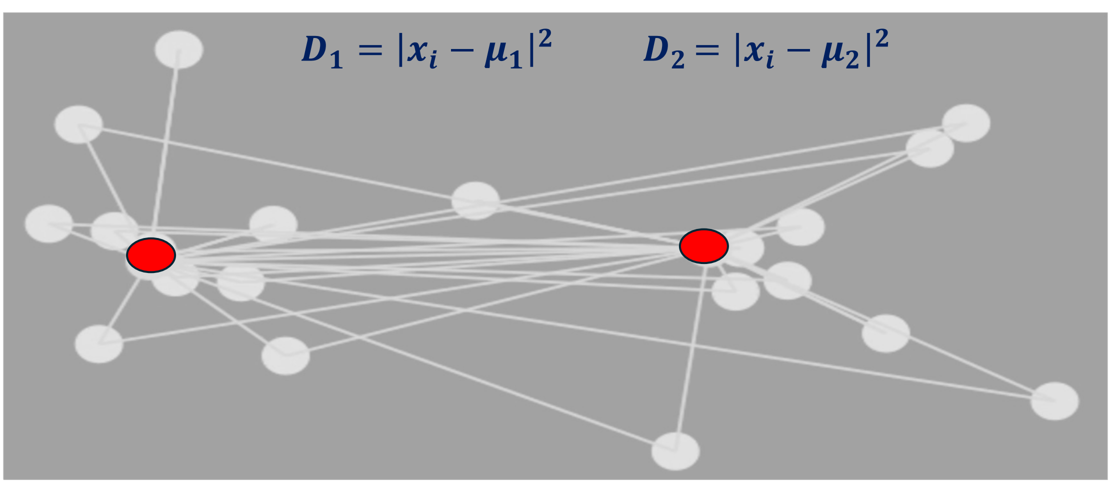
Pseudo-centroids are chosen randomly



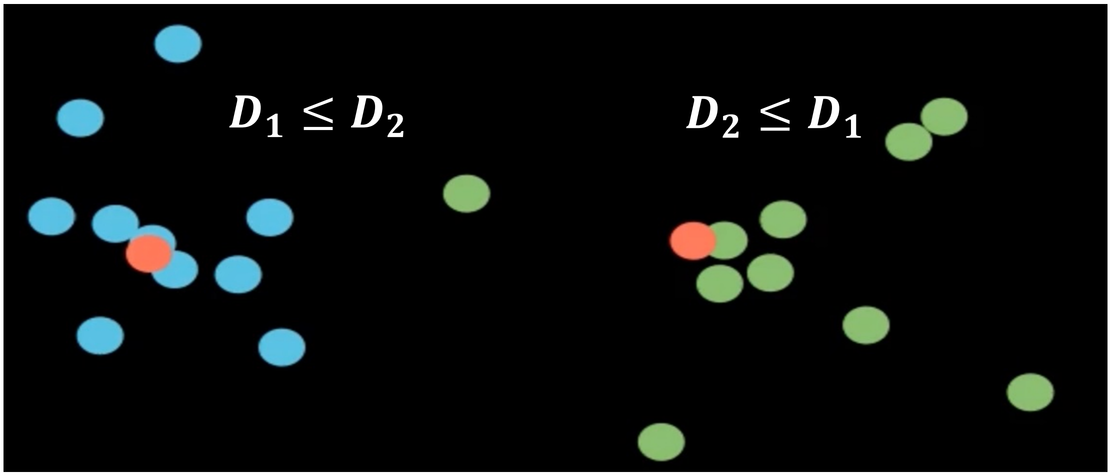
Distances from each data point to the 1st random pseudo-centroid are computed



Distances from each data point to the 2nd random pseudo-centroid are computed and compared with the distances from the 1st random centroid

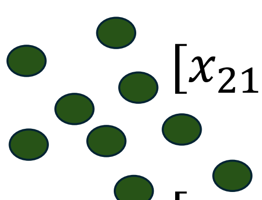


The point in the middle is green since it is closer to the pseudo-centroid representing the green cluster



Recompute the (real) centroids of the clusters

$[x_{11}, x_{12}, \dots, x_{1n}]$



$[x_{21}, x_{22}, \dots, x_{2n}]$

$[x_{k1}, x_{k2}, \dots, x_{kn}]$

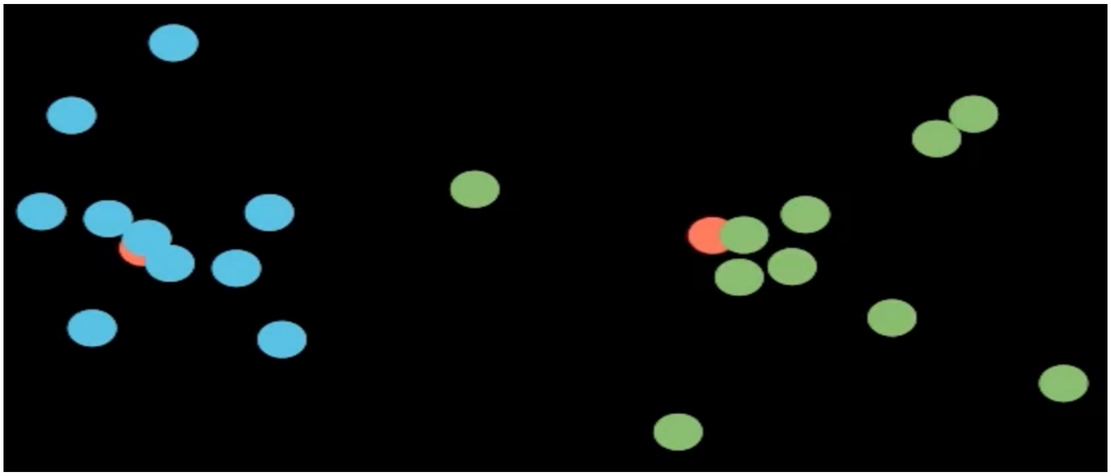
$[x_{m1}, x_{m2}, \dots, x_{mn}]$

$\boldsymbol{m\ data\ points}$

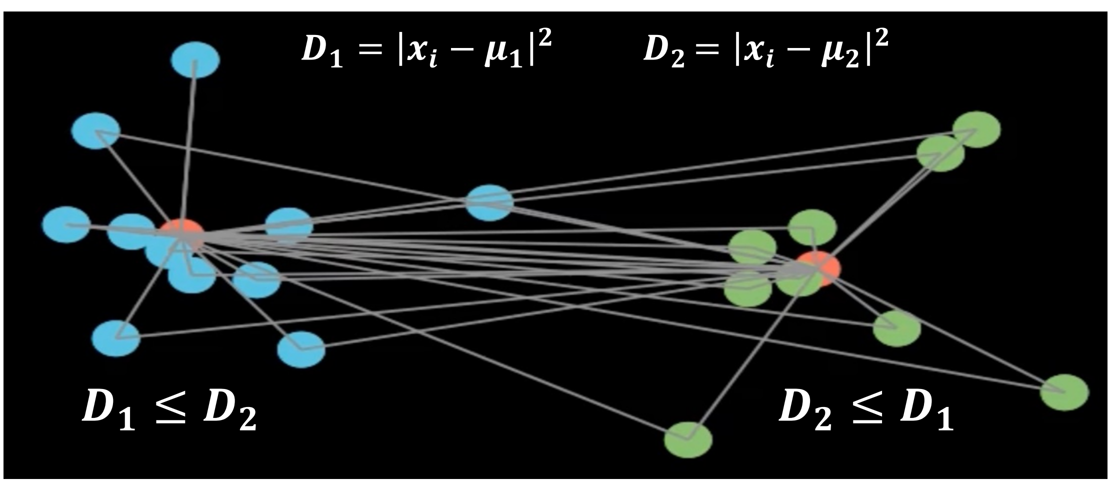
$$\mu = \frac{1}{m}\left(\sum_{i=1}^{m} x_{i1}, \sum_{i=1}^{m} x_{i2}, \dots, \sum_{i=1}^{m} x_{in}\right)$$

$$\vec{\boldsymbol{\mu}}_c = \frac{\mathbf{1}}{|\boldsymbol{c}|}\left(\sum_{i=1}^{|c|} \vec{\boldsymbol{x}}\right)$$

The pseudo-centroids move inside the clusters toward their real centers to become real centroids



Distances are again computed from both (real) centroids and compared => point in the middle now changes to blue

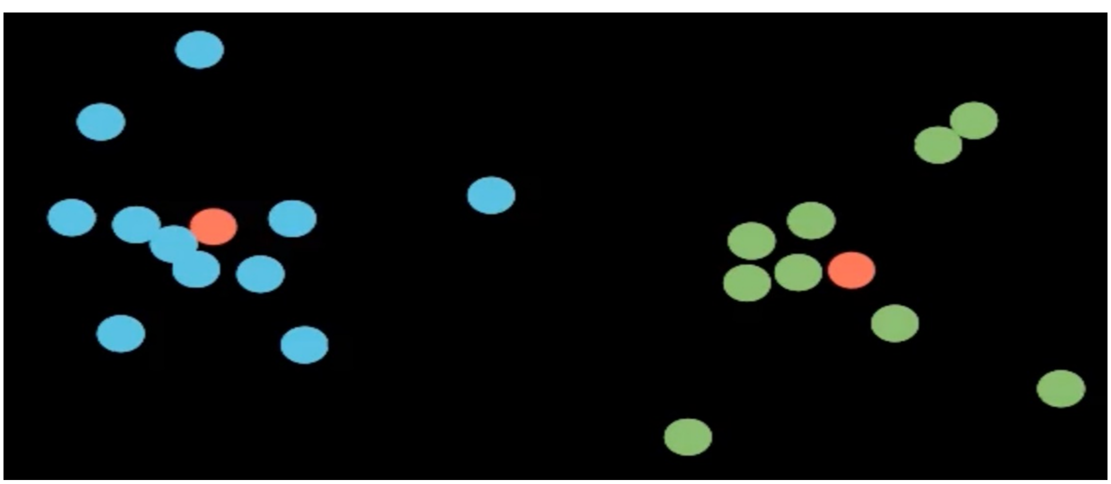


What happens to the objective function with this change to the point in the middle?

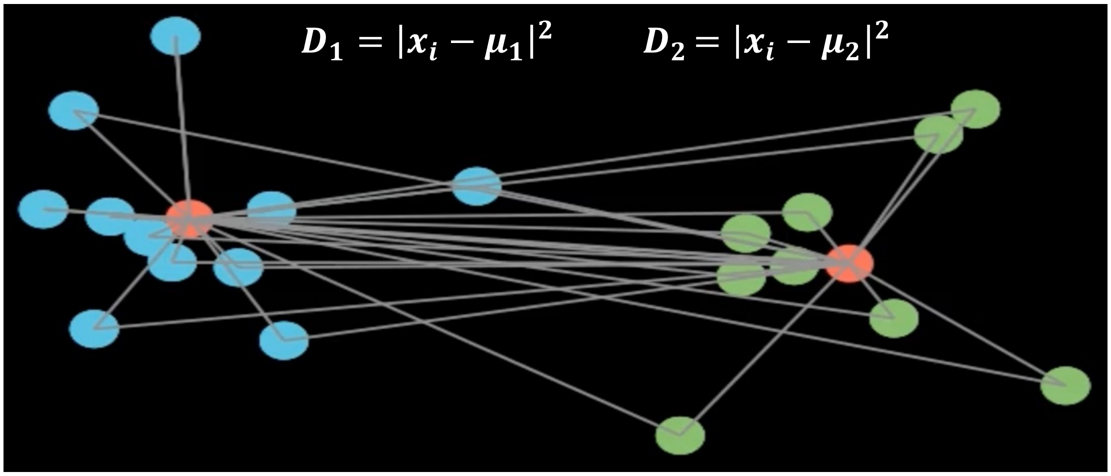
$$Z = \sum_{j=1}^{K} \sum_{n \in C_j} \left| x_n - \mu_j \right|^2$$

It reduces!

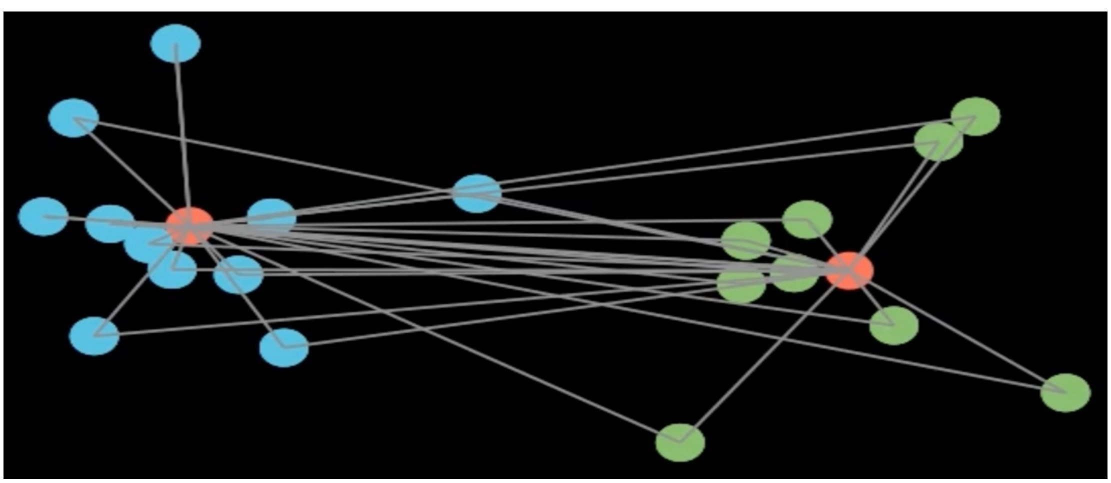
New centroids move further inside



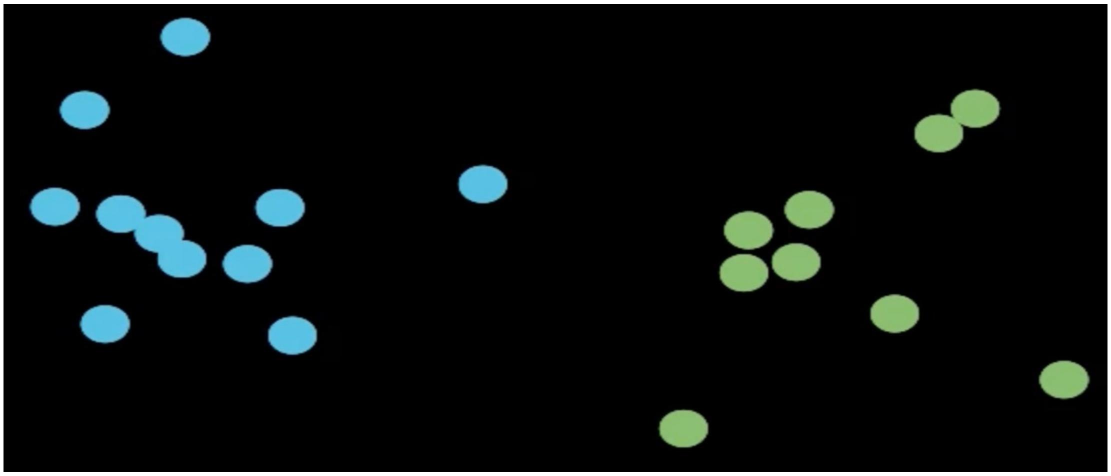
Distances from the new centroids are computed

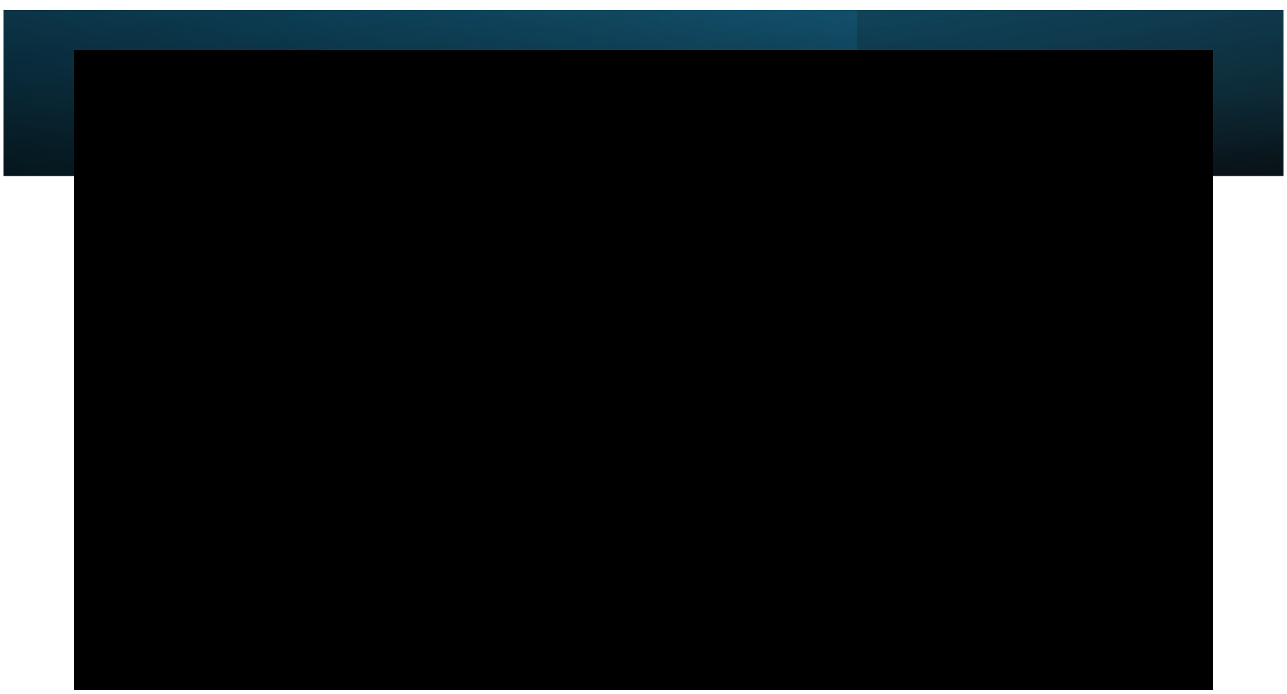


Based on the distances, the points do not change clusters => convergence



Final clusters after convergence





K-Means Algorithm

Initialization

- Randomly select K initial centroids: $\mu_1, \mu_2, \ldots, \mu_K$
- Initial selection critically impacts final clustering

Repeat

1. Assignment Step

For each data point x:

Assign to closest centroid: $argmin_j ||x - \mu_j||^2$

2. Update Step

Recalculate centroids:

$$\mu_i = \left(\frac{1}{|C_i|}\right) \Sigma_{x \in C_i} x$$

Until

Stopping Conditions:

Centroids no longer move significantly $||\mu_{t+1} - \mu_t|| < \varepsilon$

(or) No change in cluster assignments

Illustration; 3 Features, k=2; Iteration 1, Step 1: Assignment

x1	x2	x3
1	2	3
1.5	1.8	2.5
5	8	9
8	8	7
1	0.6	1
9	11	12

Random Centroids: Centroid 1: (0, 0, 0) Centroid 2: (10, 10, 10)

x1	x2	x3	Cluster	$D_1 = \|x_i - \mu_1\|^2$	$D_2 = \|x_i - \mu_2\|^2$
1	2	3	1	3.742	13.928388
1.5	1.8	2.5	1	3.426368	13.990711
5	8	9	2	13.038405	5.477226
8	8	7	2	13.304135	4.123106
1	0.6	1	1	1.536229	15.822768
9	11	12	2	18.601075	2.449490

$$D_1 = \sqrt{(0-1)^2 + (0-2)^2 + (0-3)^2} = \sqrt{14} = 3.742$$

$$D_2 = \sqrt{(10-1)^2 + (10-2)^2 + (10-3)^2} = \sqrt{81+64+49} = \sqrt{194} = 13.928$$

Iteration 1: Update Step: Recalculate centroids.

- **Centroid 1**: Mean of (1, 2, 3), (1.5, 1.8, 2.5), (1, 0.6, 1)
= ((1+1.5+1)/3, (2+1.8+0.6)/3, (3+2.5+1)/3) = (1.167, 1.467, 2.167)
- **Centroid 2**: Mean of (5, 8, 9), (8, 8, 7), (9, 11, 12) = (7.333, 9, 9.333)

x1	x2	x3	Cluster	$D_1 = \|x_i - \mu_1\|^2$	$D_2 = \|x_i - \mu_2\|^2$
1	2	3	1	3.742	13.928388
1.5	1.8	2.5	1	3.426368	13.990711
5	8	9	2	13.038405	5.477226
8	8	7	2	13.304135	4.123106
1	0.6	1	1	1.536229	15.822768
9	11	12	2	18.601075	2.449490

Iteration 2: Assignment Step

From Iteration 1

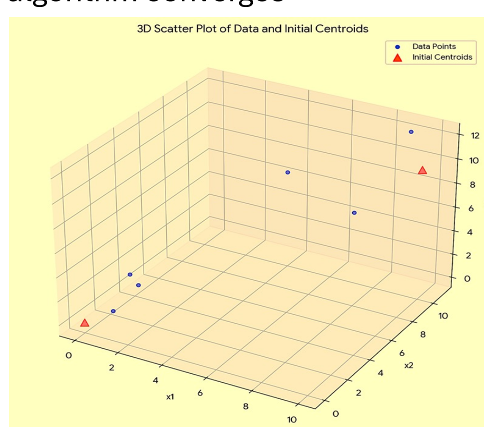
- **Centroid 1**: (1.167, 1.467, 2.167)
- **Centroid 2**: (7.333, 9, 9.333)

x1	x2	x3	C	$D_1 = \|x_i - \mu_1\|^2$	$D_2 = \|x_i - \mu_2\|^2$
1.0	2.0	3.0	1	1.003328	11.367595
1.5	1.8	2.5	1	0.577350	11.513567
5.0	8.0	9.0	2	10.201634	2.560382
8.0	8.0	7.0	2	10.617909	2.624669
1.0	0.6	1.0	1	1.462874	13.420714
9.0	11.0	12.0	2	15.777833	3.726780

From Iteration 2

- **Centroid 1**: (1.167, 1.467, 2.167)
- **Centroid 2**: (7.333, 9, 9.333)

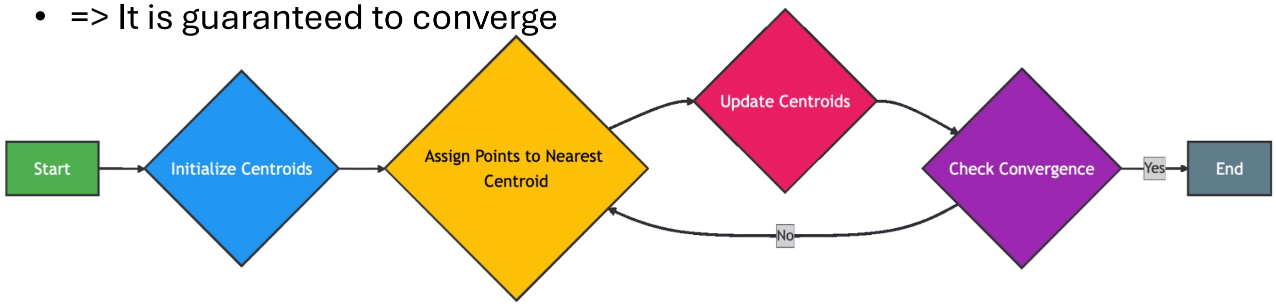
Centroids do not change => algorithm converges



K-means Flowchart

- EM is proven to converge
- K-means is is a special case of EM
- => It is guaranteed to converge

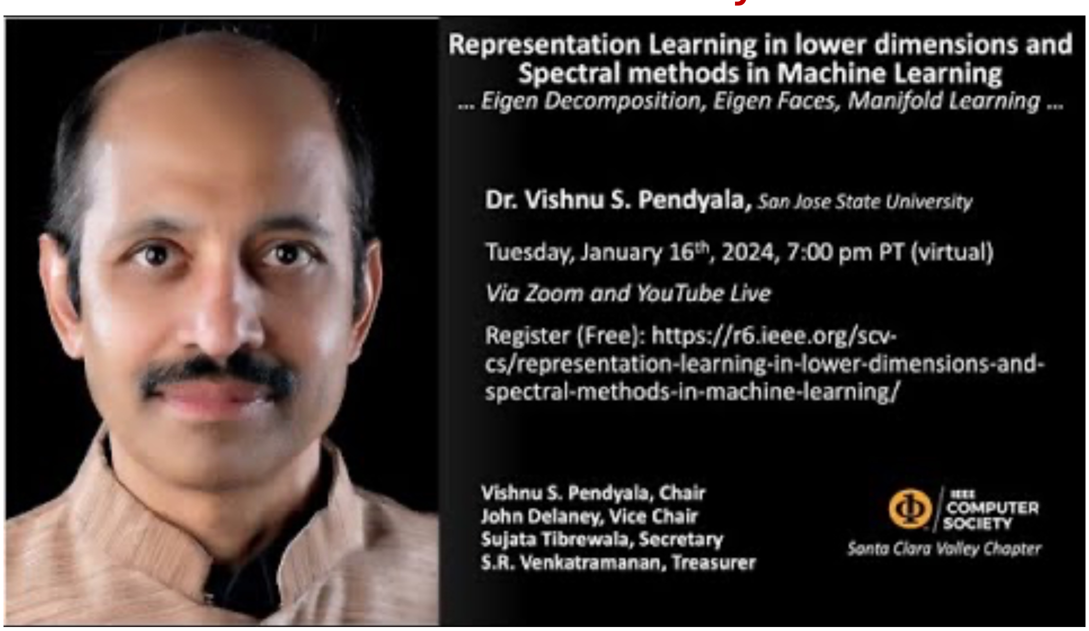
Maximization



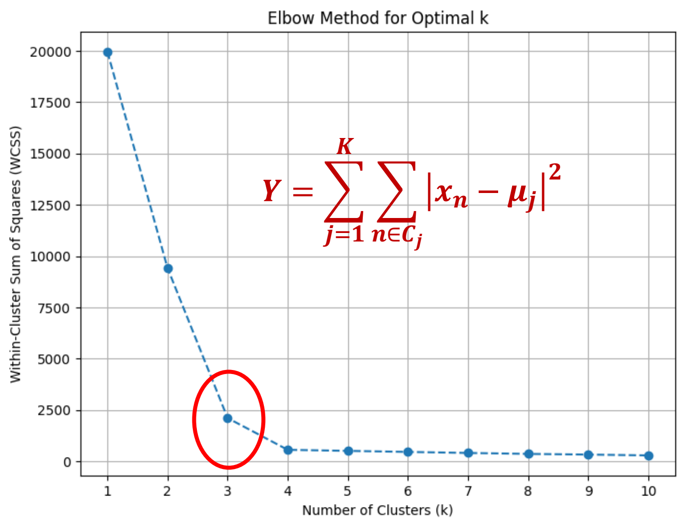
Expectation

- But convergence does not imply optimality!
- **Bad seeds can result in bad clusters / slow convergence**

How do we know how many clusters?



Determining K: Elbow method



K	WCSS
1	19953.76914827803
2	9416.214004352274
3	2110.4125218953286
4	564.9141808210253
5	513.0329042790798
6	462.0960007878173
7	411.3489318727926
8	365.3555992995912
9	332.0364571015475
10	291.5608624106775

Computational complexity of K-Means

Computationally intensive steps:

The algorithm loops for

i: #iterations until convergence

K: #clusters for each cluster

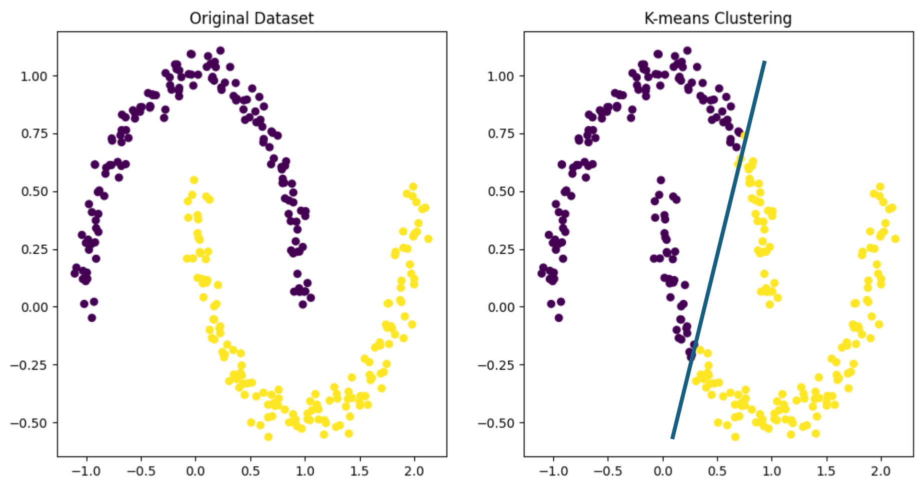
n: #datapoints to compute distance of each point from k centroids

d: #dimensions for computing Euclidean distance from centroid

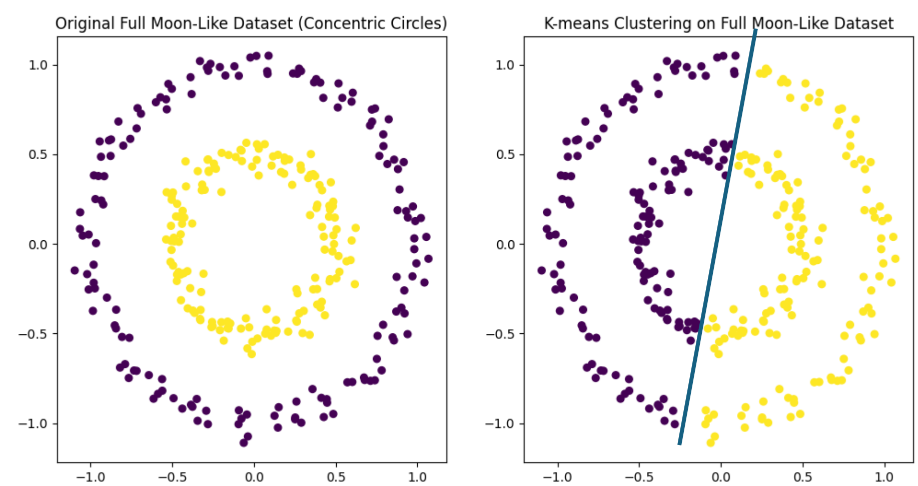
E.g.: $\sqrt{(10-1)^2 + (10-2)^2 + (10-3)^2}$

Therefore, time complexity: O(n * K * d * i)

K-Means can only draw linear boundaries!



K-Means can only draw linear boundaries!



DBSCAN – Key insights

- Clusters data points that are closely packed (density).
- Points that are in low-density regions are flagged as noise.

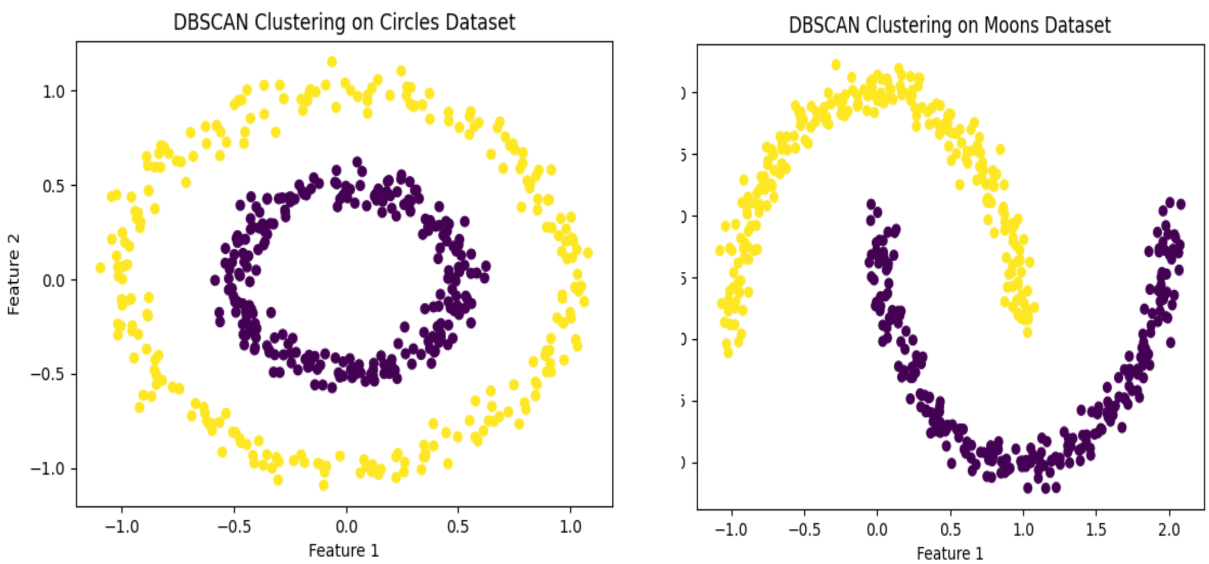
Simple algorithm based on two hyperparameters to define "dense"

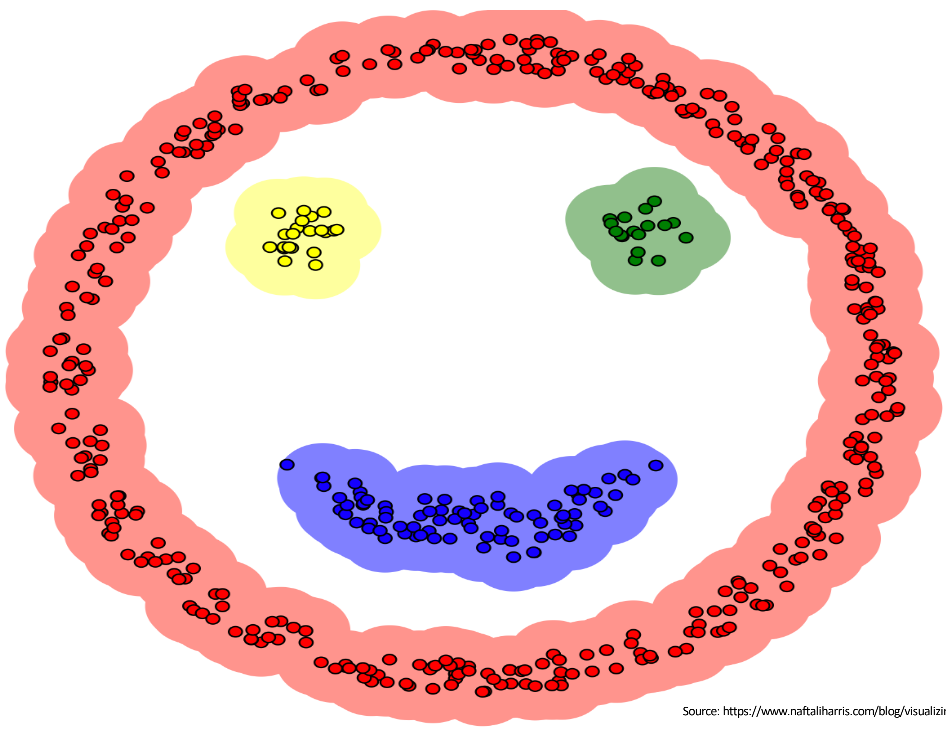
- **Epsilon (ε): M**aximum distance between two points for them to be considered as neighbors.
- **MinPts:** Minimum number of points required to form a dense region within a radius **ε**.

Some Applications:

- Geospatial Data: Identifying regions with high population density.
- Anomaly Detection: Detecting fraud or irregular patterns in data.
- Image Segmentation: Identifying regions of interest in images.

DBSCAN can form clusters of varied shapes





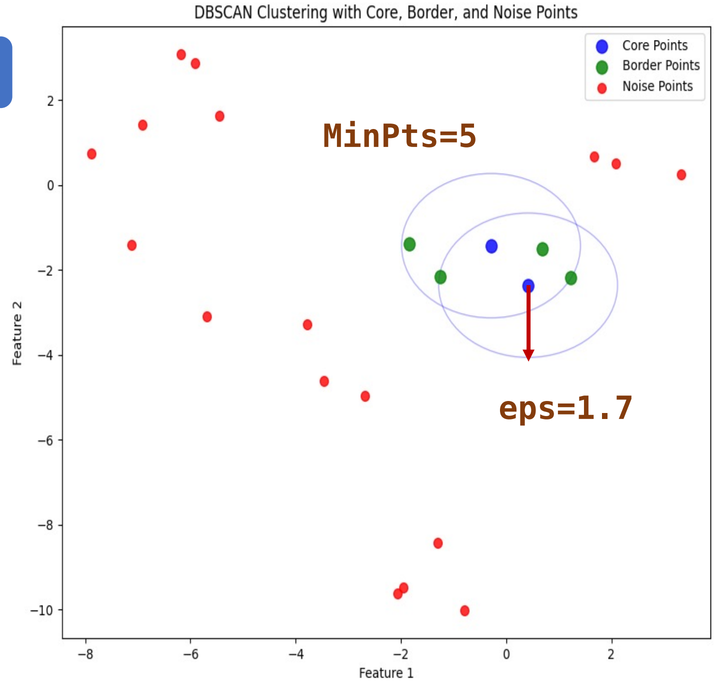
56

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

Core, Border, and Noise Points

- **Core Points**: Points with at least MinPts neighbors within ε and will be included in a cluster.
- **Border Points**: Points that are reachable from a core point but do not have enough neighbors to be core points.
- **Noise Points**: Points that are neither core nor border points, often considered outliers.

DBSCAN Clustering with Core, Border, and Noise Points



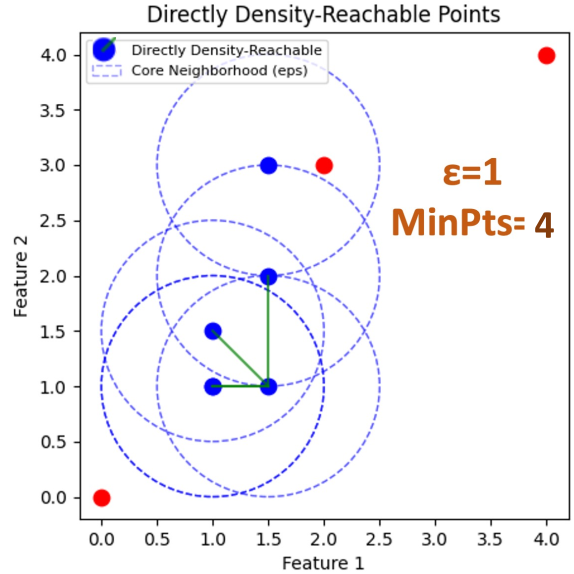
Directly density-reachable

A point **p** is **directly density-reachable** from point **q** w.r.t. ε and MinPts if

- **p is within the ε-neighborhood of q**: $distance(p, q) \leq \varepsilon$ *and*
- **q is a core point**: $|\{p' \in D \mid distance(q, p') \leq \varepsilon\}| \geq MinPts$

Directly density reachability is not symmetric!

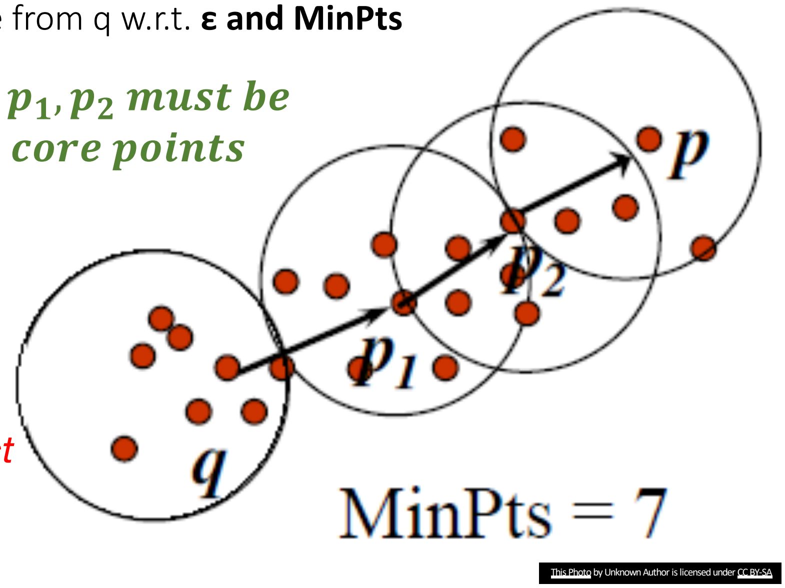
Directly Density-Reachable Points



There is a chain of points that are directly density-reachable, starting from **q** and ending at **p =>** p is density reachable from q w.r.t. **ε and MinPts**

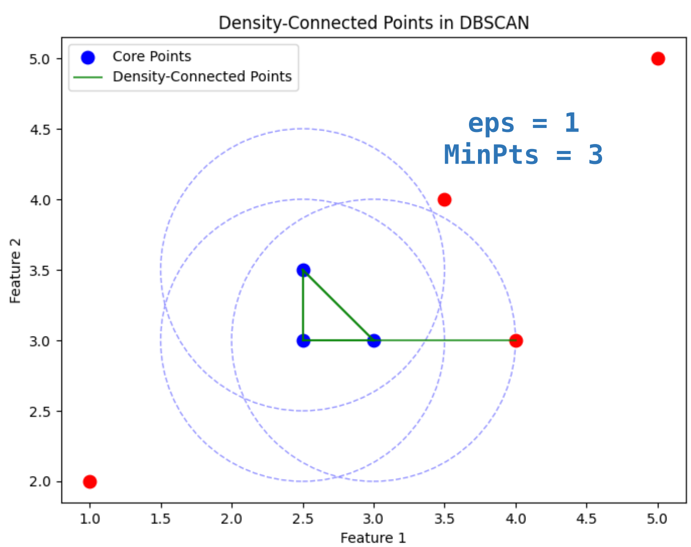
Density Reachable Points

q, p_1, p_2 must be core points



Density reachability is also not symmetric because direct density-reachability is not!

Density Connected



p and q are said to be **density-connected** w.r.t. eps and MinPts if there exists a point v such that:

- **p is directly density-reachable from v**
- **q is directly density-reachable from v**

Density connected is symmetric!

DBSCAN – the algorithm - initialization

Input: A set of data points, along with two key parameters:

- ε (epsilon): The maximum radius of the neighborhood around a point.
- minPts: The minimum number of points required to form a dense region (cluster).

Output: A set of clusters, with some points possibly marked as noise.

For each data point, DBSCAN classifies it into one of three categories:

- Core Point: A point that has at least minPts points (including itself) within its ε-neighborhood.
- Border Point: A point that has fewer than minPts points within its ε-neighborhood but is within the ε-neighborhood of a core point.
- Noise Point: A point that is neither a core point nor a border point.

DBSCAN Algorithm - Forming clusters

If a point is a core point, a new cluster is started, and all points in its ε-neighborhood are added to this cluster.

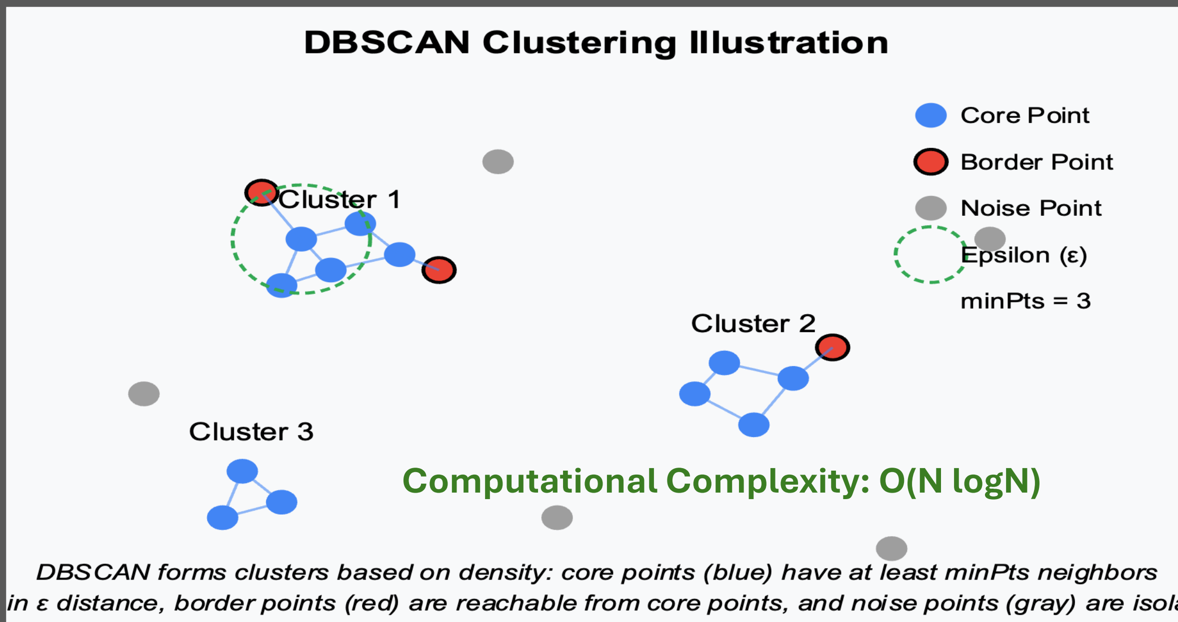
If any of those neighboring points are core points themselves, their ε-neighborhoods are recursively processed.

For each core point's ε-neighborhood, recursively expand the cluster by checking whether the neighboring points are core points or border points.

Border points are assigned to the cluster of the core point that expanded them.

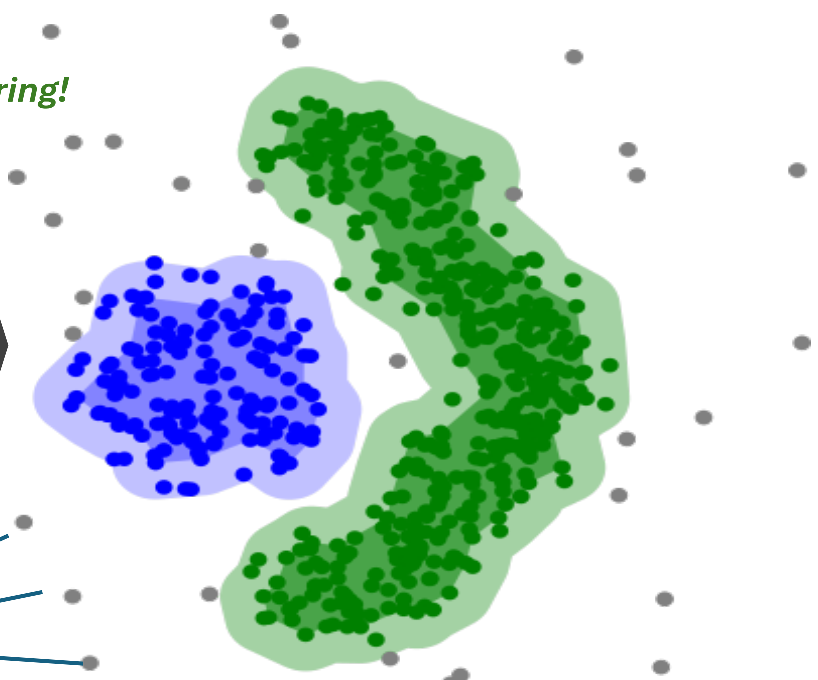
Points that are neither core points nor border points are labeled as noise.

The algorithm stops when all points have been processed (either assigned to a cluster or marked as noise).



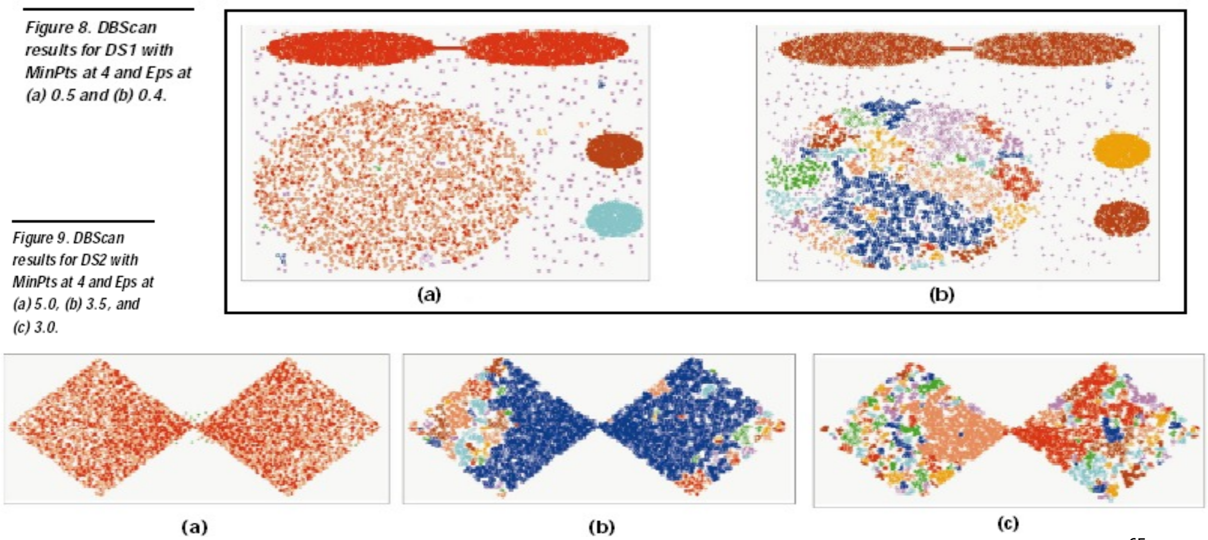
Noise resistant Clustering!

DBSCAN used in preprocessing for removing noise



Noise

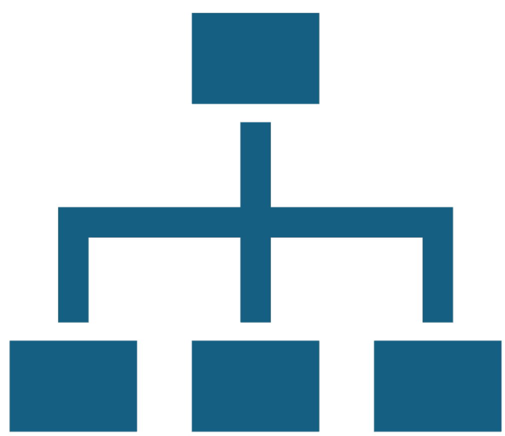
DBSCAN is sensitive to hyperparameters and it is hard to choose the right ones



65

Source: Data Mining: Concepts and Techniques (3rd ed.)

Hierarchical Clustering



Hierarchical Clustering – Key Insights

- Hierarchical clustering organizes data into a hierarchy of nested clusters, visualized as a dendrogram.
- Widely used in Biological sciences, Gene clustering, and Taxonomy creation such as in web catalogs.
- No need to predefine the number of clusters.
- Flexible: Any number of clusters can be obtained by cutting the dendrogram at the desired level.
- Quadratic complexity => Computationally intensive for large datasets

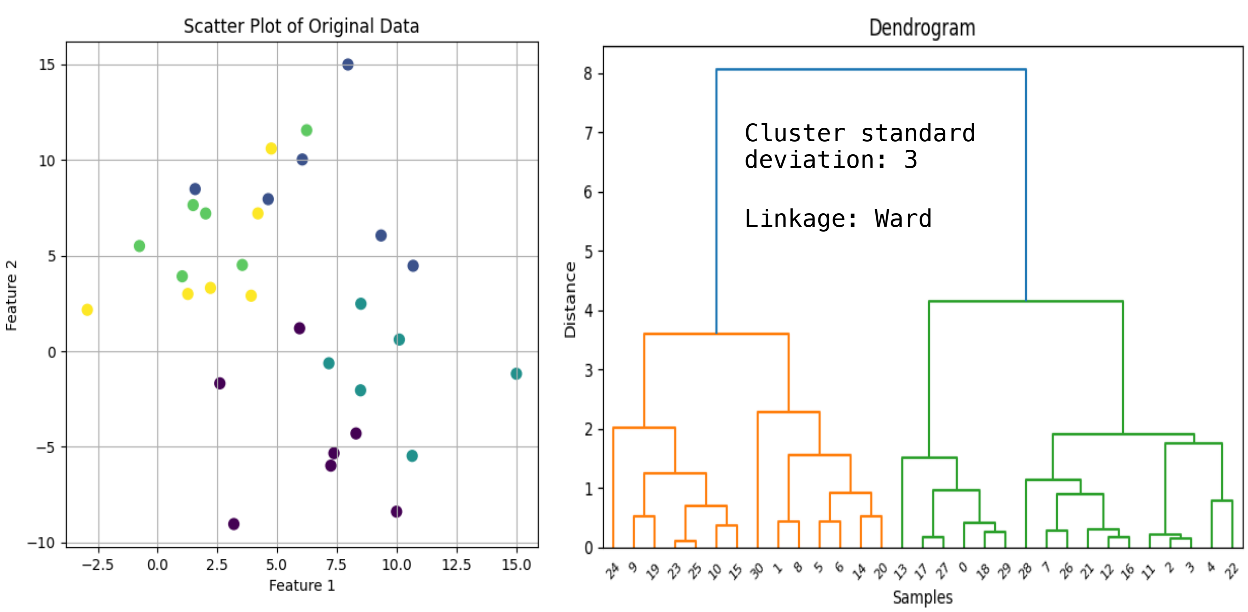
Agglomerative (Bottom-Up):

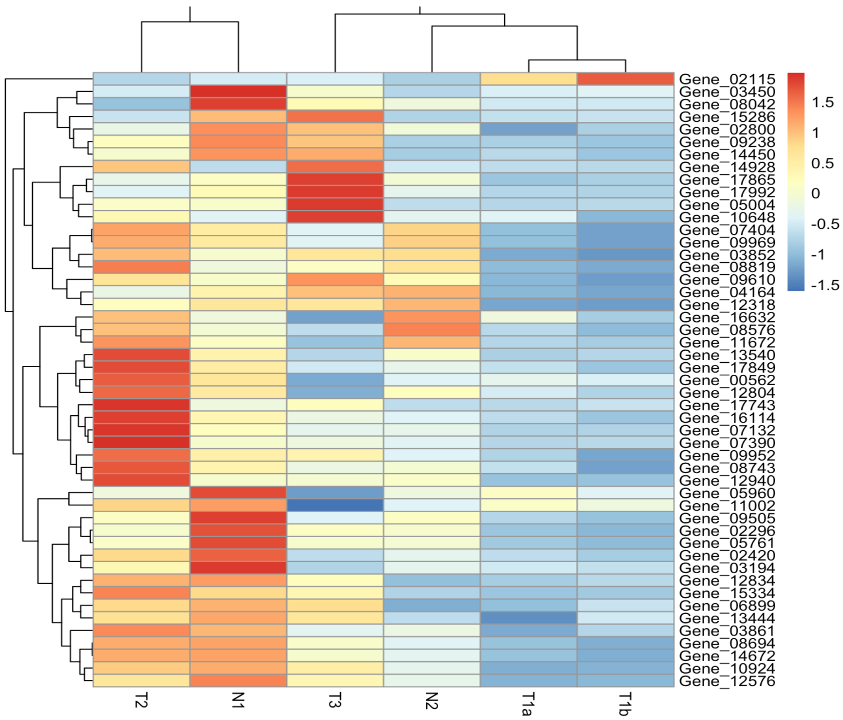
- Start with each point as its own cluster.
- Iteratively merge the closest clusters until one cluster remains.

Divisive (Top-Down):

- Start with one all-inclusive cluster.
- Recursively split clusters into smaller groups based on dissimilarity.

Dendrogram for 5 clusters





Visualizing gene expression analysis using a heatmap

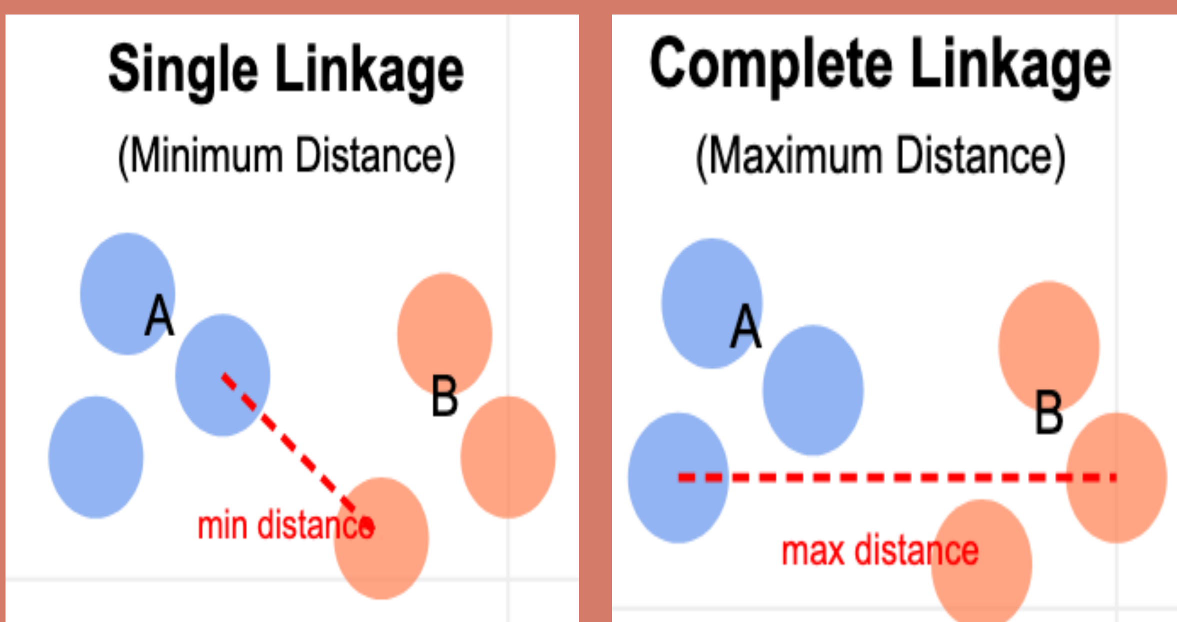
Divisive Clustering Algorithm

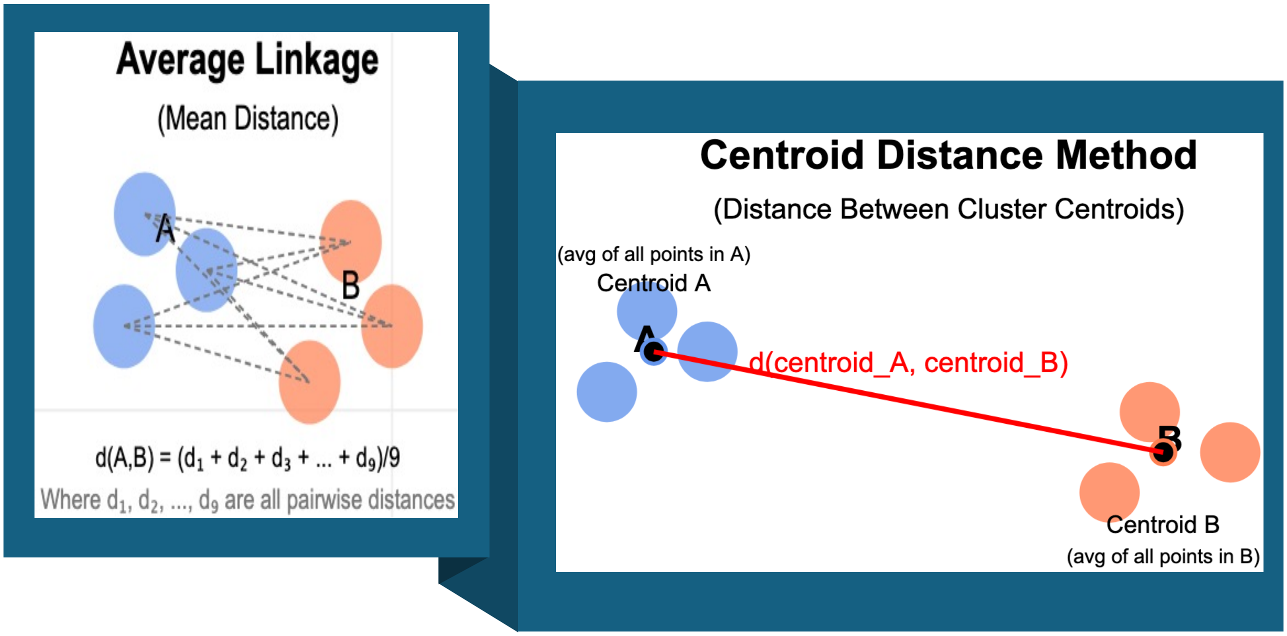
- Start with all points in one cluster.
- Recursively split clusters based on dissimilarity.
- The sequence of splits can be shown using a dendrogram.
- Horizontal cuts to the dendrogram define the number of clusters desired
- Less commonly used due to higher computational demands

Agglomerative Clustering Algorithm

1. Compute the distance matrix between data points.
2. Let each point be a cluster.
3. Repeat:
   1. **Merge the two closest clusters.**
   2. **Update the distance matrix.**
4. Stop when only one cluster remains

- **Distance Metrics**:
  - Single linkage (minimum distance).
  - Complete linkage (maximum distance).
  - Average linkage
  - Distance between centroids
  - Ward's method





Ward's method

Step 2: Calculate Within-Cluster Sum of Squares

For cluster A:

$ESS(A) = \Sigma ||x - centroid\_A||^2$

$= d_1^2 + d_2^2 + d_3^2$

For cluster B:

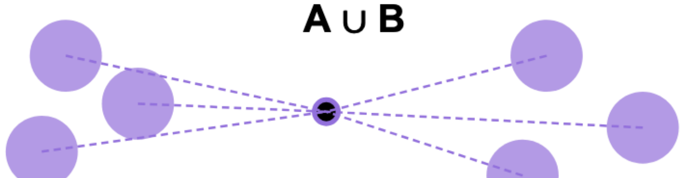
$ESS(B) = \Sigma ||x - centroid\_B||^2$

$= d_4^2 + d_5^2 + d_6^2$

Total Error Sum of Squares = ESS(A) + ESS(B)

Step 3: Calculate Error for Hypothetical Merged Cluster

A ∪ B



Merged Centroid = (|A|·centroid_A + |B|·centroid_B)/(|A|+|B|)

$ESS(A \cup B) = \Sigma ||x - centroid\_merged||^2$

$= d_1'^2 + d_2'^2 + d_3'^2 + d_4'^2 + d_5'^2 + d_6'^2$

Step 4: Calculate Ward's Distance

Ward's Criterion = ESS(A∪B) - [ESS(A) + ESS(B)]

= Increase in within-cluster sum of squares

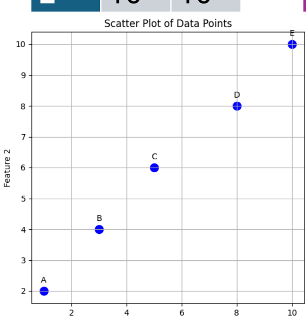
Agglomerative Clustering - Illustration

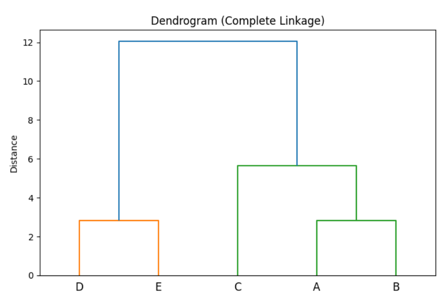
Dataset

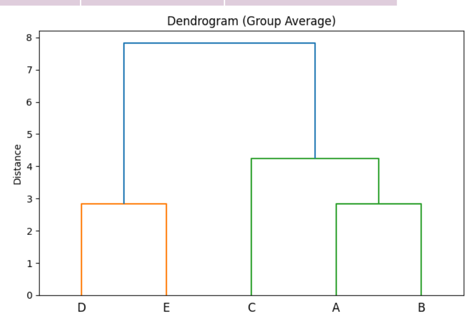
	x1	x2
A	1	2
B	3	4
C	5	6
D	8	8
E	10	10

Proximity Matrix (Euclidean Distance)

	A	B	C	D	E
A	0.000000	2.828427	5.656854	9.219544	12.041595
B	2.828427	0.000000	2.828427	6.403124	9.219544
C	5.656854	2.828427	0.000000	3.605551	6.403124
D	9.219544	6.403124	3.605551	0.000000	2.828427
E	12.041595	9.219544	6.403124	2.828427	0.000000







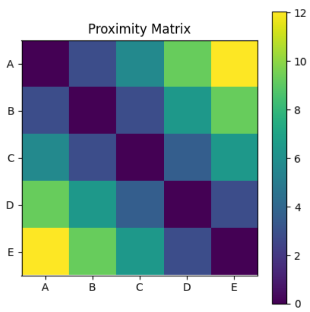
Agglomerative Clustering - Illustration

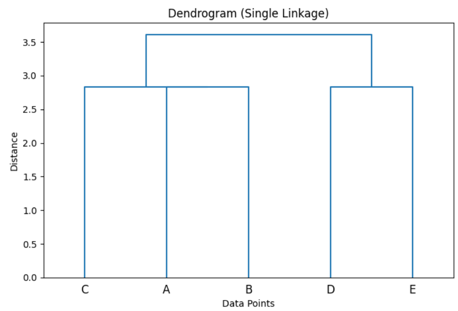
Dataset

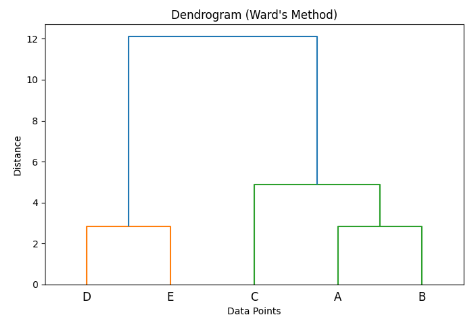
	x1	x2
A	1	2
B	3	4
C	5	6
D	8	8
E	10	10

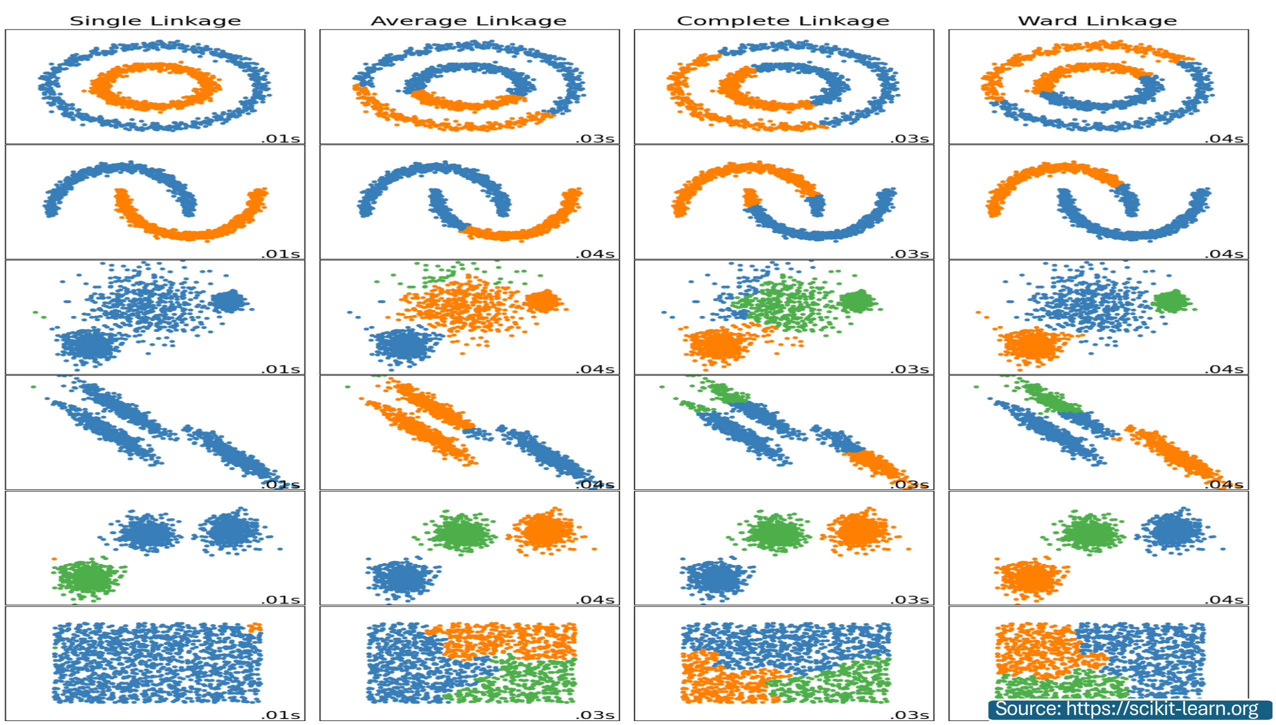
Proximity Matrix (Euclidean Distance)

	A	B	C	D	E
A	0.000000	2.828427	5.656854	9.219544	12.041595
B	2.828427	0.000000	2.828427	6.403124	9.219544
C	5.656854	2.828427	0.000000	3.605551	6.403124
D	9.219544	6.403124	3.605551	0.000000	2.828427
E	12.041595	9.219544	6.403124	2.828427	0.000000

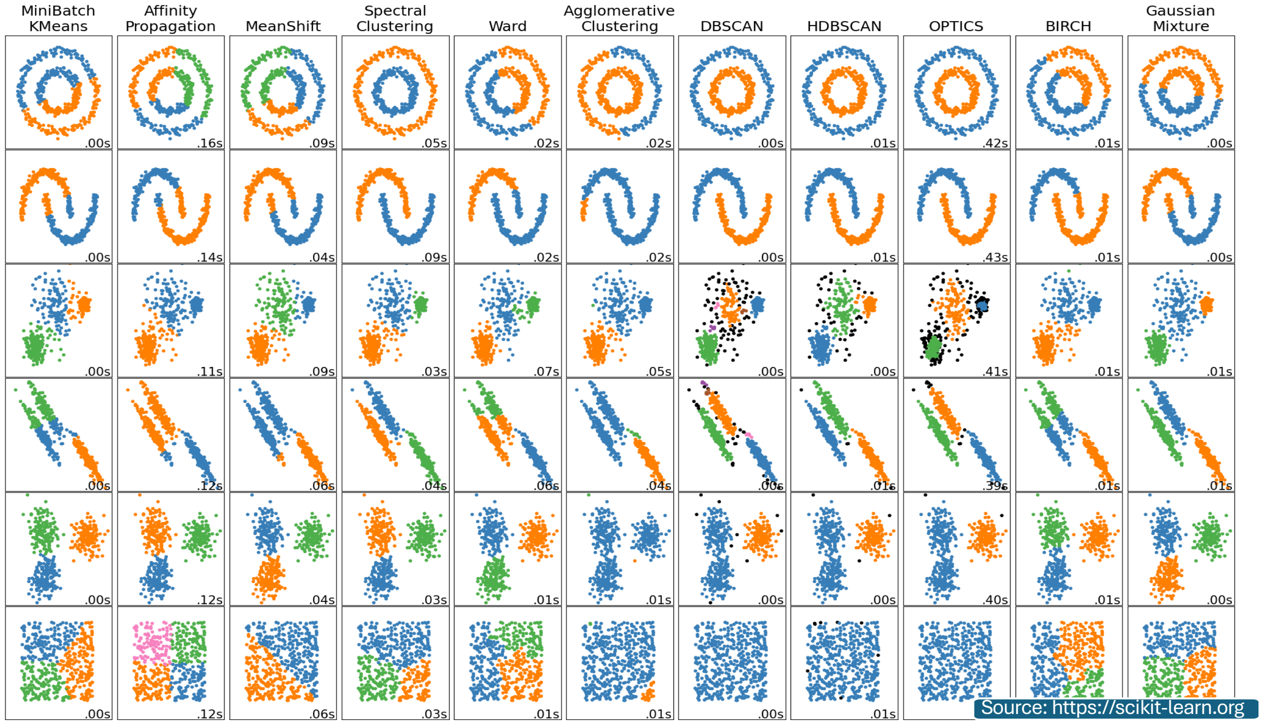


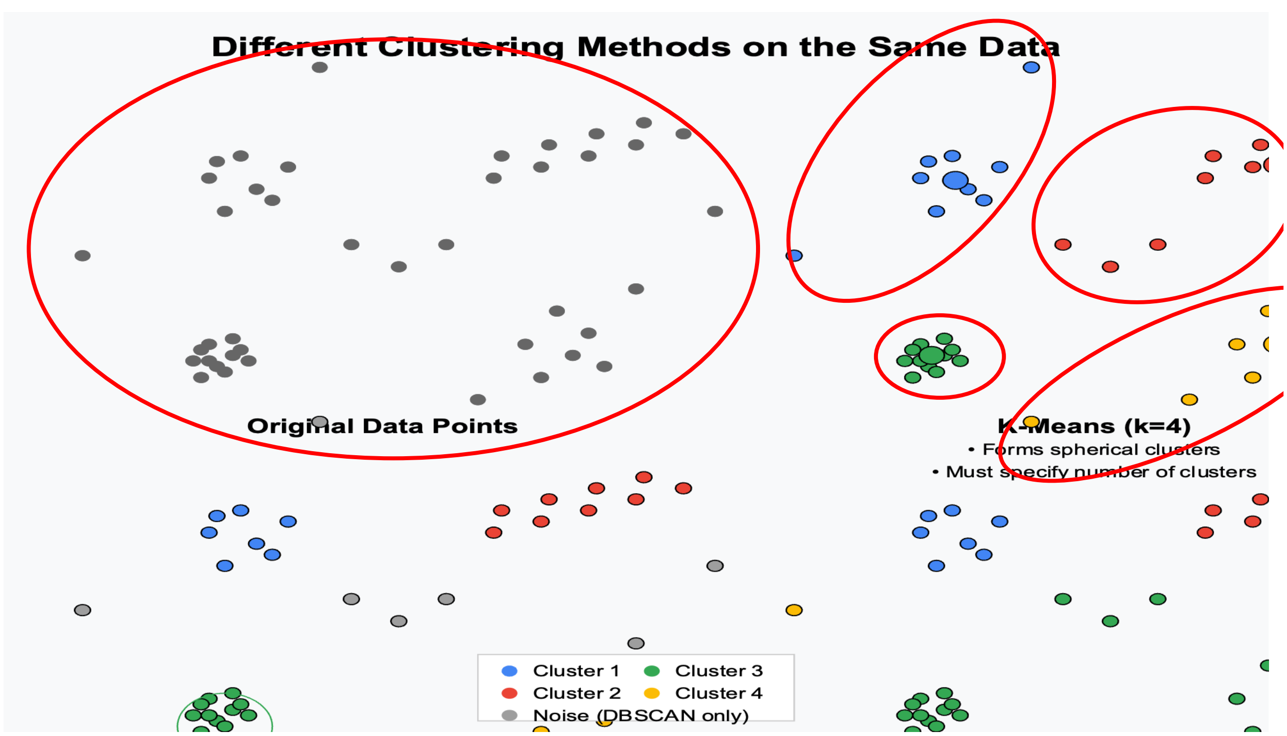






Source: https://scikit-learn.org


MiniBatch KMeans
Affinity Propagation
MeanShift
Spectral Clustering
Ward
Agglomerative Clustering
DBSCAN
HDBSCAN
OPTICS
BIRCH
Gaussian Mixture
Source: https://scikit-learn.org


Different Clustering Methods on the Same Data
Original Data Points
K-Means (k=4)
• Forms spherical clusters
• Must specify number of clusters
Cluster 1
Cluster 3
Cluster 2
Cluster 4
Noise (DBSCAN only)

How do we know which cluster arrangement is the best?

No evaluation metric is perfect; need to depend on heuristics

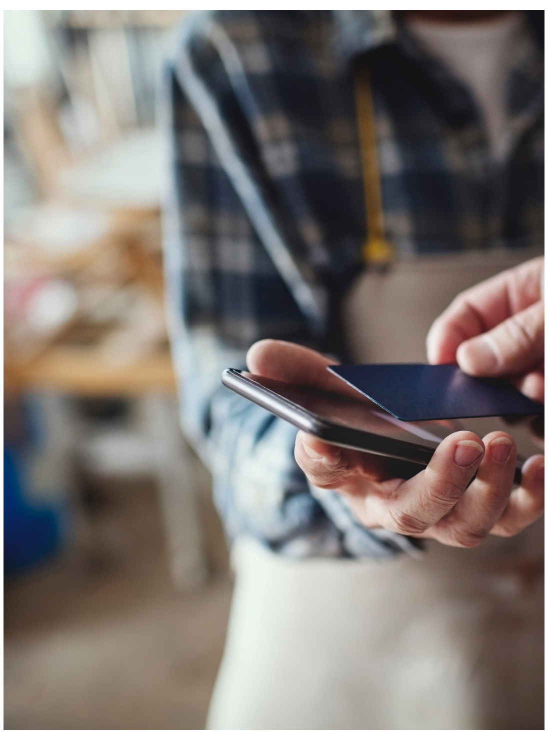
Evaluating clusters

- Metrics: Sum of intra-cluster distances
- But what matters is the impact of the clustering on business / application needs!
- Final authority in evaluation is the human user

Silhouette Score to evaluate the quality of clustering

$Silhouette\ score$ ‘s’ = $\frac{b-a}{\max(a,b)}$ where range is [-1, 1] and

- **a:** The average distance between a point and all other points in its own cluster (intra-cluster distance).
- **b:** The average distance between a point and all the points in the nearest cluster it does not belong to (inter-cluster distance).
- Often used for selecting the optimal number of clusters
- S close to 1: Data point is well-clustered, far from other clusters.
- S close to 0: Data point is on the border of clusters, unclear which cluster it belongs to.
- S close to -1: Data point is probably in the wrong cluster.
- Sensitive to the shape of clusters. May not perform well for non-convex clusters.



Stay in touch!

https://twitter.com/VishnuPendyala

https://www.facebook.com/vishnu.pendyala

https://www.instagram.com/vishnupendyala/

https://www.threads.net/@vishnupendyala

https://www.linkedin.com/in/pendyala/

Questions

and answers