Faculty of Computer Science | Data Analysis | Moscow 2025

Lecture 5
Cluster Analysis

Lecturer: Alisa Melikyan, amelikyan@hse.ru, PhD,
Associate Professor of the School of Software Engineering

Cluster Analysis

Cluster analysis (or clustering) is an unsupervised learning technique that groups similar objects into clusters based on a measure of similarity. Items in each cluster are similar in some ways to each other and dissimilar to those in other clusters.

Steps of cluster analysis

1. Select a **distance measure** between clusters.
2. Select a **clustering algorithm.**
3. Determine the **number of clusters.**
4. Validate the analysis.

Distance measure

The most straightforward and generally accepted way of computing distances between objects in a multi-dimensional space is to compute Euclidian distances (for continuous variables). There are other specialized measures for categorical variables.

Euclidean Distance

$$D_{ij} = \sqrt{\sum_{k=1}^{n} \left(x_{ki} - x_{kj} \right)^2}$$

Euclidian distance is the geometric distance between two objects (or cases).

D_{ij} – distance between cases *i* and *j*

x_{ki} value of variable x_k for case i

Problems:

- Different measures = different weights

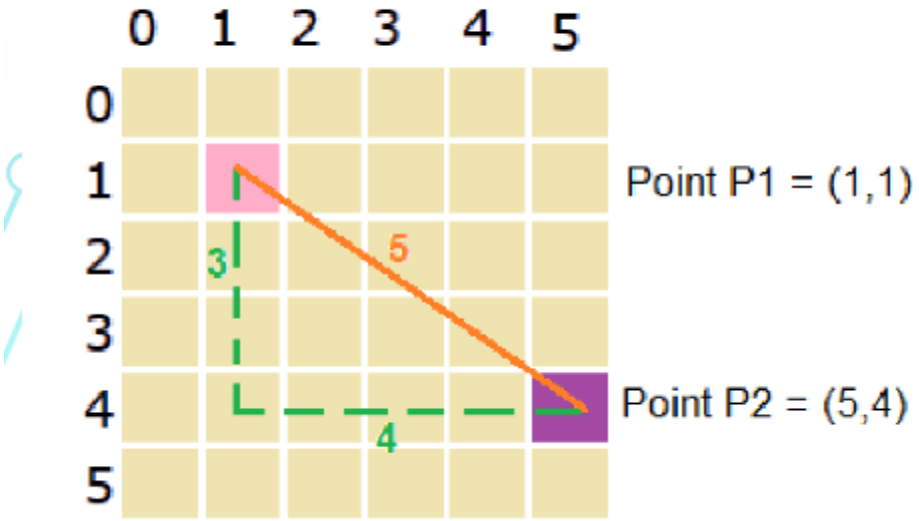
  Solution: *Standardization*

- Correlation between variables (double counting)

  Solution: *PCA before cluster analysis*

Manhattan Distance

$$\rho(a,b) = |x_1 - x_2| + |y_1 - y_2|$$



Euclidean distance = $\sqrt{(5-1)^2 + (4-1)^2} = 5$

Manhattan distance = $|5-1| + |4-1| = 7$

Different clustering procedures

- **Hierarchical cluster analysis**:
  - agglomerative (start from n clusters, to get to 1 cluster),
  - divisive (initially, all the points in the dataset belong to one cluster and split is performed recursively as we move down the hierarchy).
- **"K-Family" clustering:** k-means, k-medians, k-modes, k-prototypes.
- **DBSCAN.**

Hierarchical cluster analysis

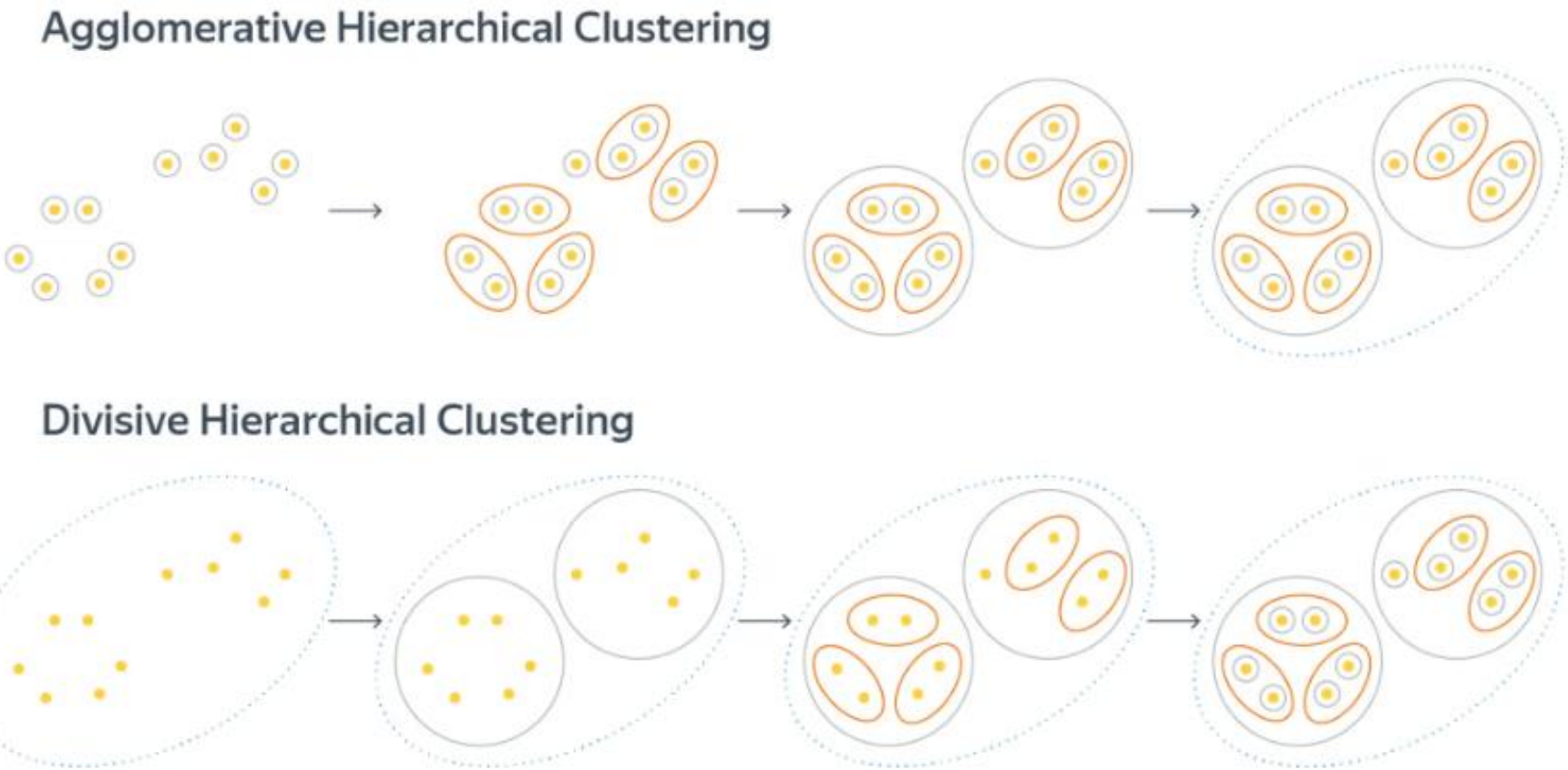
Is recommended for a relatively small data set if the purpose is to easily examine solutions with increasing numbers of clusters. First, each case is treated as an individual cluster. Clusters are then merged based on a criterion specific to the chosen method. So, we begin with as many clusters as there are cases and end up with just one cluster containing all cases. By inspecting the progression of cluster merging it is possible to isolate clusters of cases with high similarity.

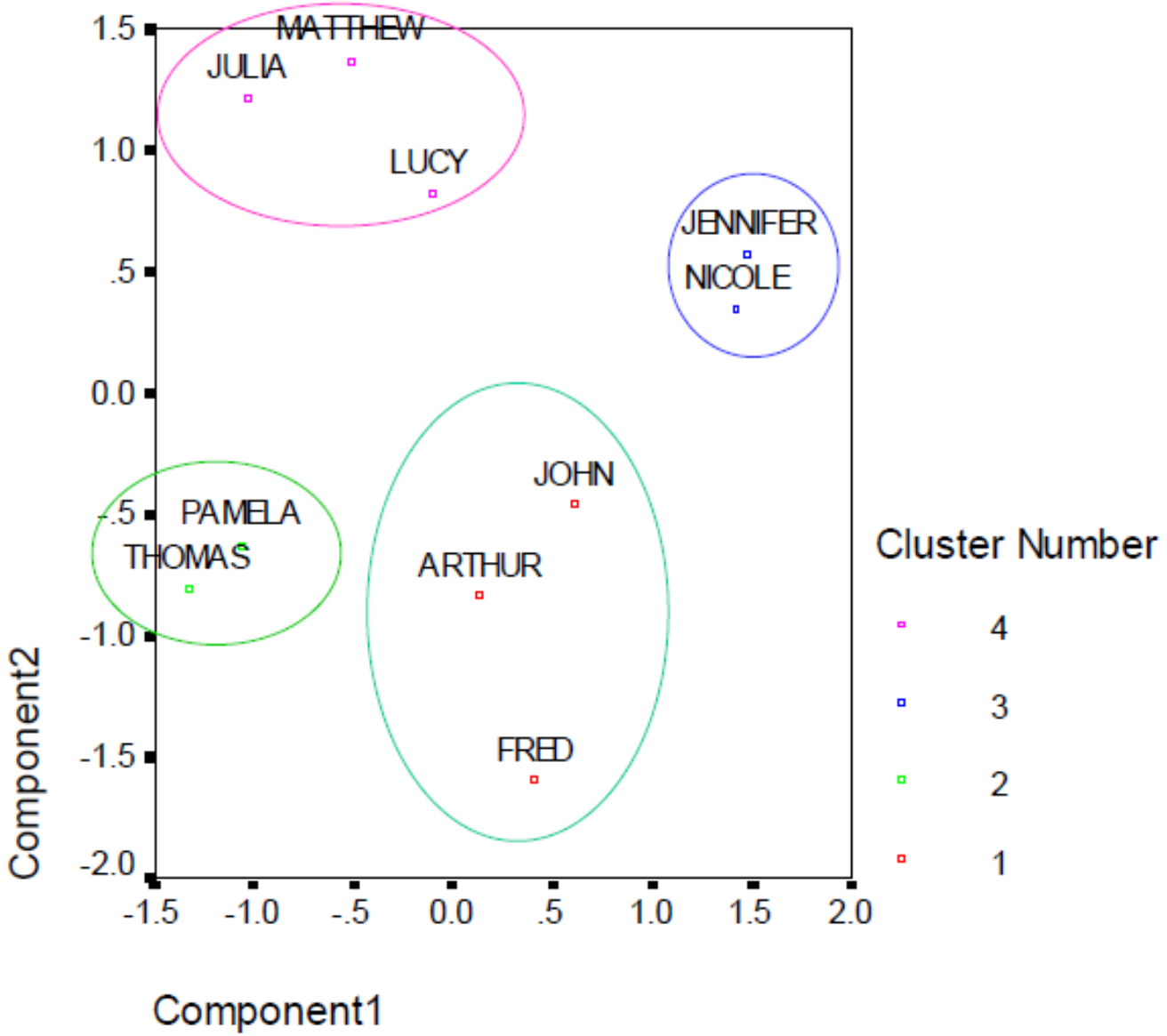
Hierarchical cluster analysis

To form clusters using a hierarchical cluster analysis, you must select:

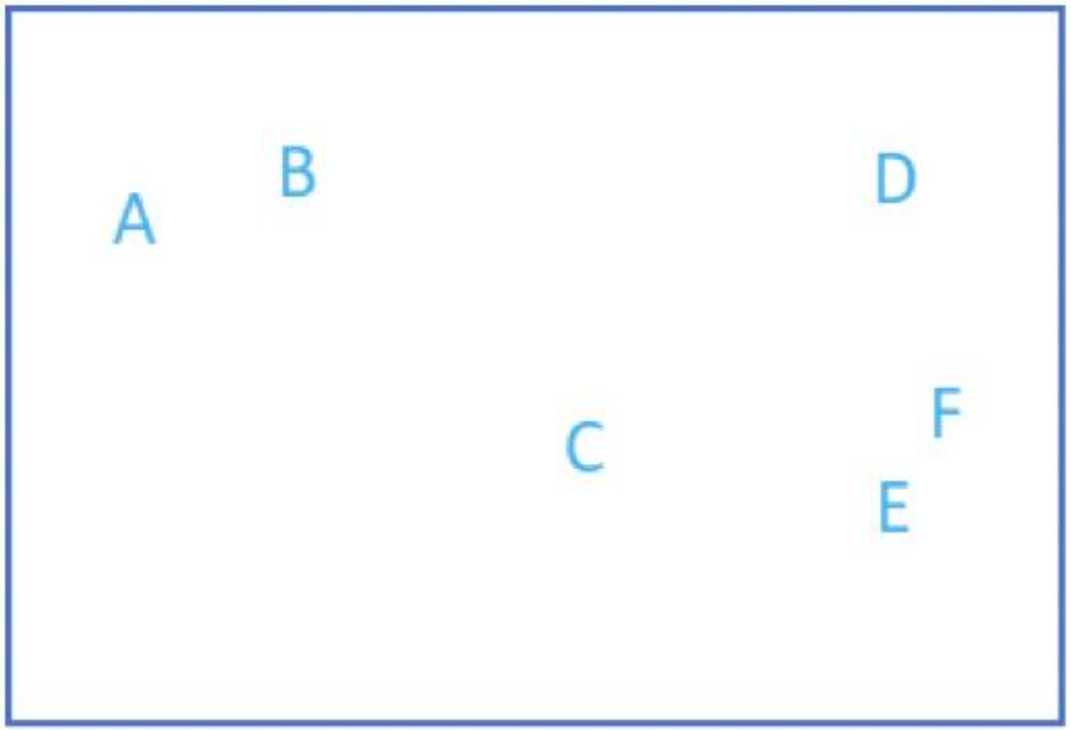
1. A criterion for determining similarity or distance between cases;
2. A criterion for determining which clusters are merged at successive steps;
3. The number of clusters you need to represent your data.

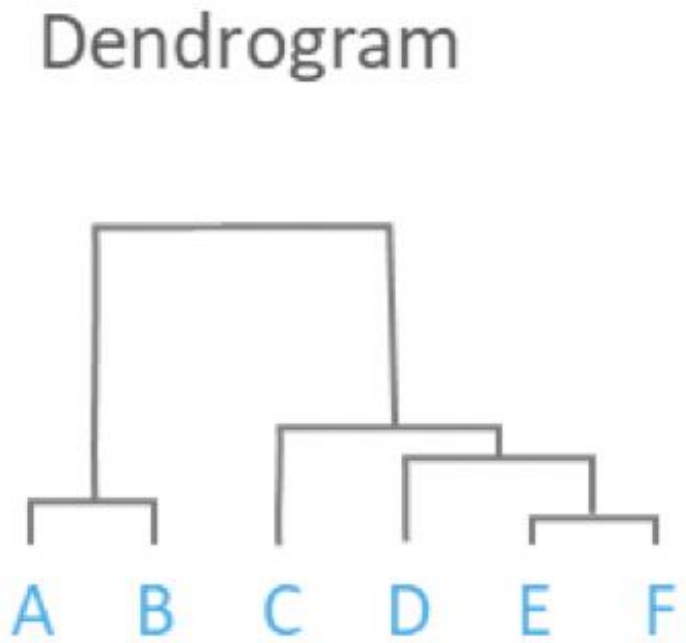
Hierarchical cluster analysis

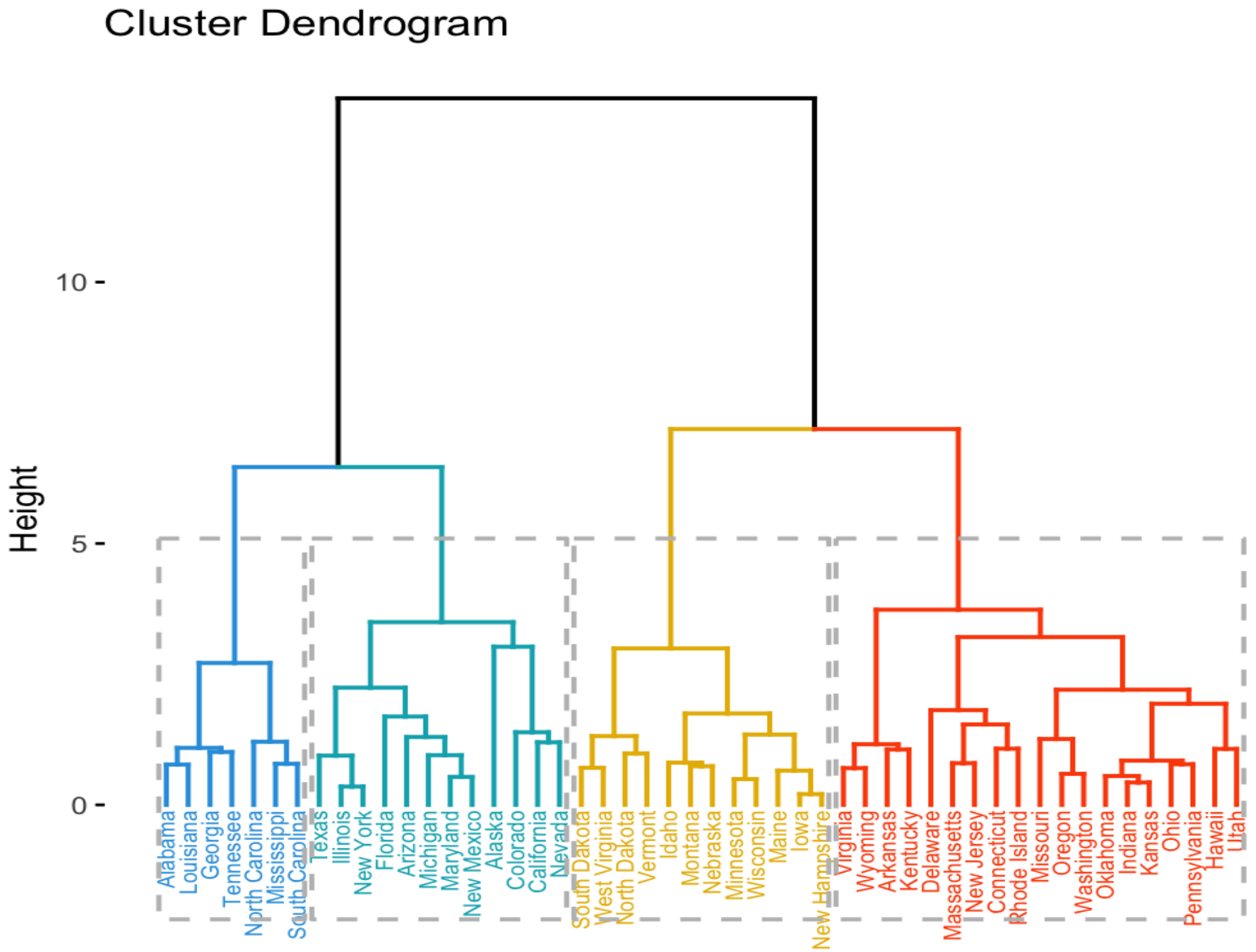


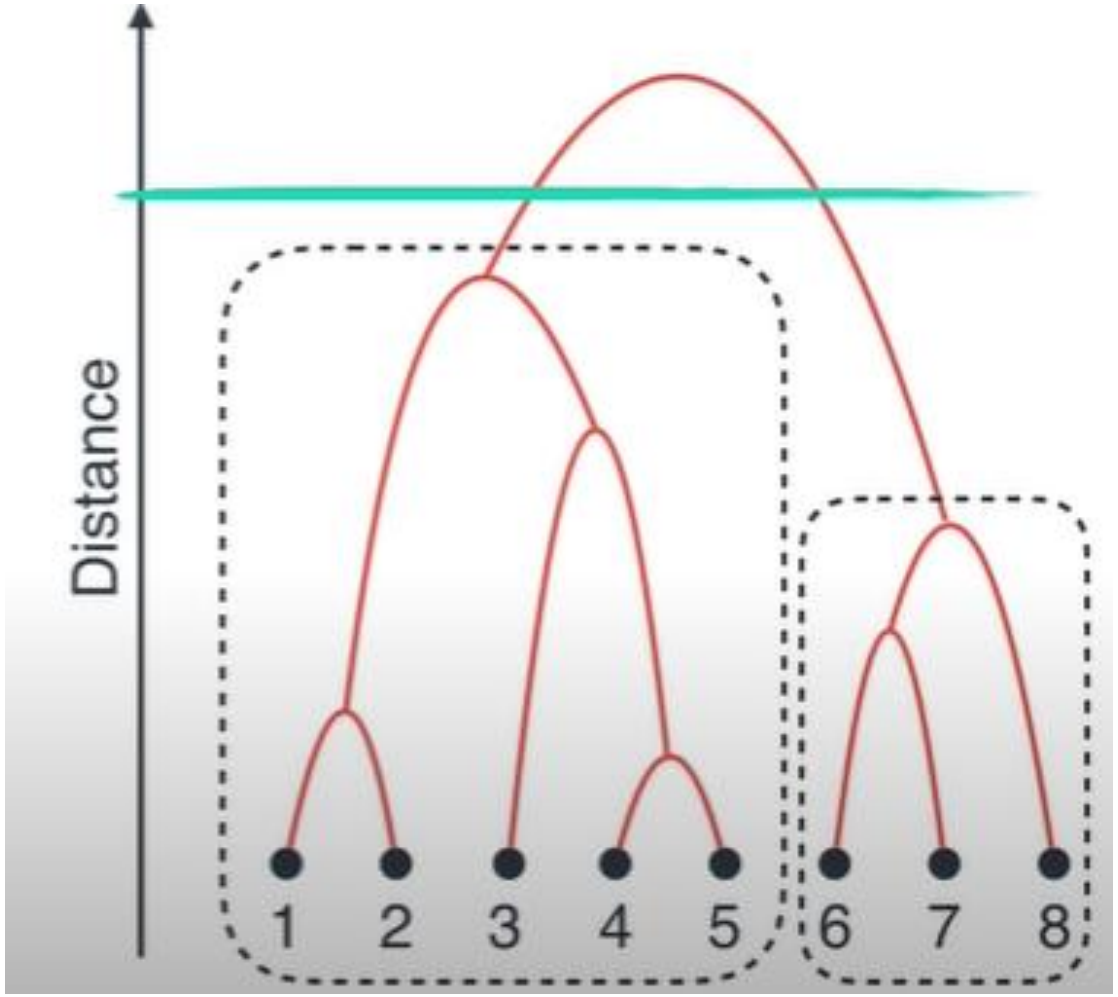

MATTHEW
JULIA
LUCY
JENNIFER
NICOLE
JOHN
PAMELA
THOMAS
ARTHUR
FRED
1.5
1.0
.5
0.0
-.5
-1.0
-1.5
-2.0
-1.5
-1.0
-.5
0.0
.5
1.0
1.5
2.0
Component2
Component1
Cluster Number
4
3
2
1

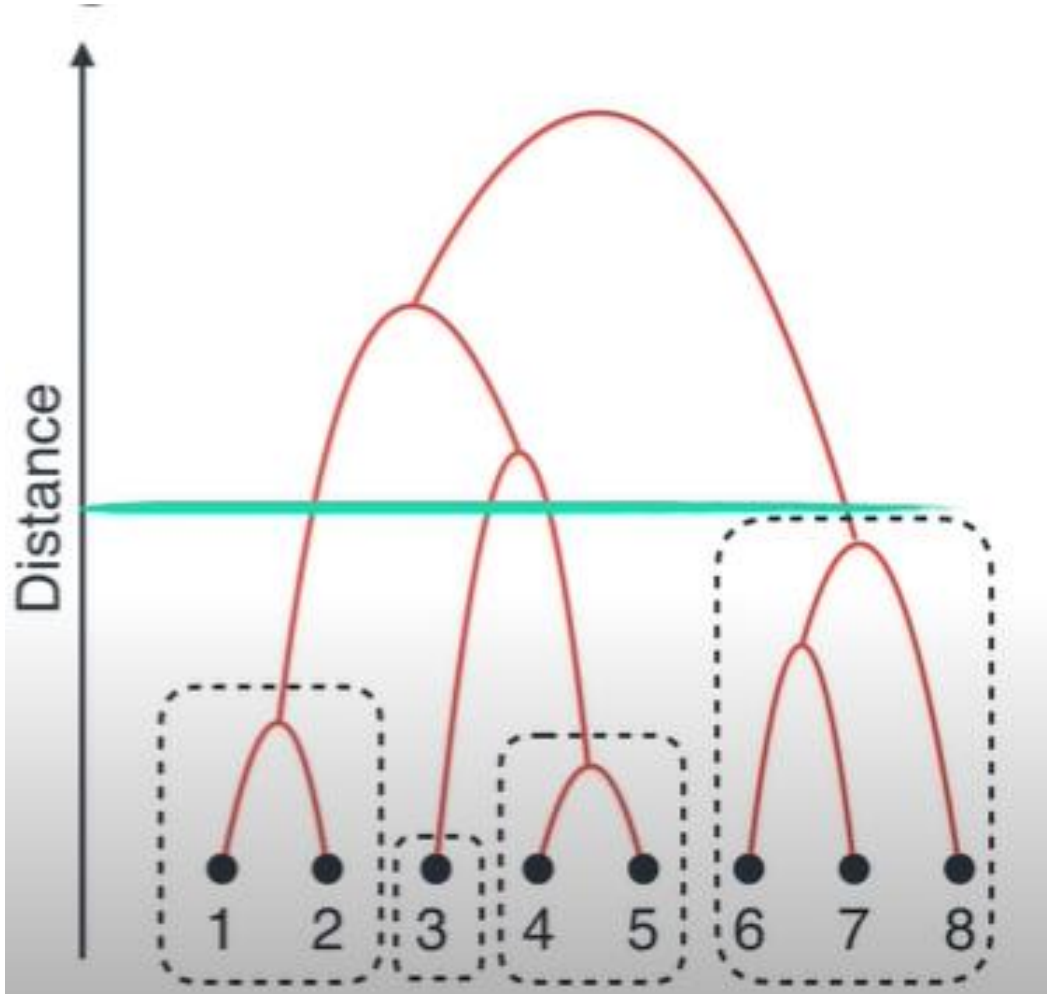
Dendrogram shows the forks (or links) between cases, and its structure gives us clues as to which cases form coherent clusters.






Cluster Dendrogram
Height
10
5
0
Alabama
Louisiana
Georgia
Tennessee
North Carolina
Mississippi
South Carolina
Texas
Illinois
New York
Florida
Arizona
Michigan
Maryland
New Mexico
Alaska
Colorado
California
Nevada
South Dakota
West Virginia
North Dakota
Vermont
Idaho
Montana
Nebraska
Minnesota
Wisconsin
Maine
Iowa
New Hampshire
Virginia
Wyoming
Arkansas
Kentucky
Delaware
Massachusetts
New Jersey
Connecticut
Rhode Island
Missouri
Oregon
Washington
Oklahoma
Indiana
Kansas
Ohio
Pennsylvania
Hawaii
Utah


Distance
1
2
3
4
5
6
7
8


Distance
1
2
3
4
5
6
7
8

Cluster Methods

There are several procedures for combining clusters. The linkage methods are all based on similar principle: there is a chain of similarity leading to whether or not a case is added to a cluster. The rules governing this chain differ from one linkage method to another.

Cluster Methods

Linkage methods

- Single linkage (minimum distance)
- Complete linkage (maximum distance)
- Average linkage (average distance)

Centroid method

- The distance between two clusters is defined as the difference between the centroids (cluster averages)

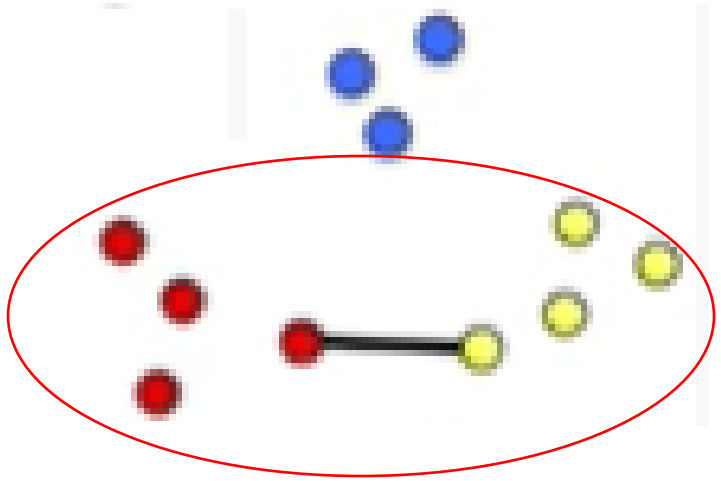
Ward's method

1. Compute sum of squared distances within clusters
2. Aggregate clusters with the minimum increase in the overall sum of squares

Nearest neighbor / Single-linkage

Defines the distance between two clusters as the distance between closest elements in clusters. It will connect the clusters that are closest to each cluster.

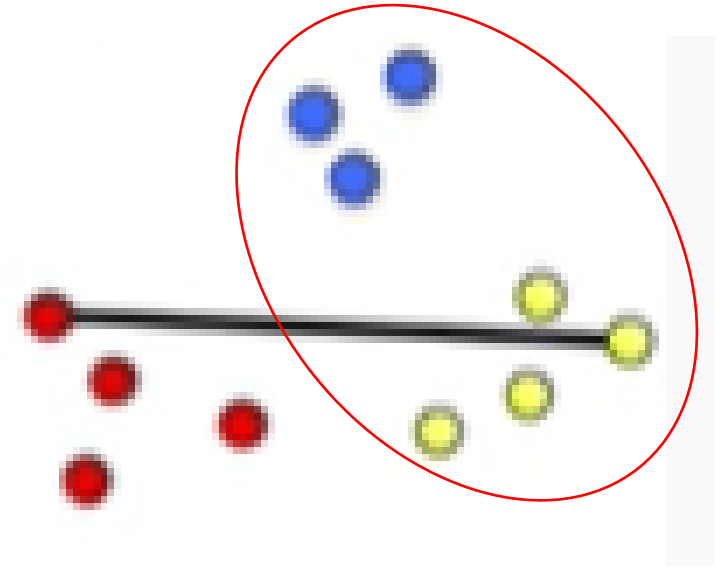
Problem: can produce long chains. It's less likely to get spherical clusters.



Furthest neighbor / Complete-linkage

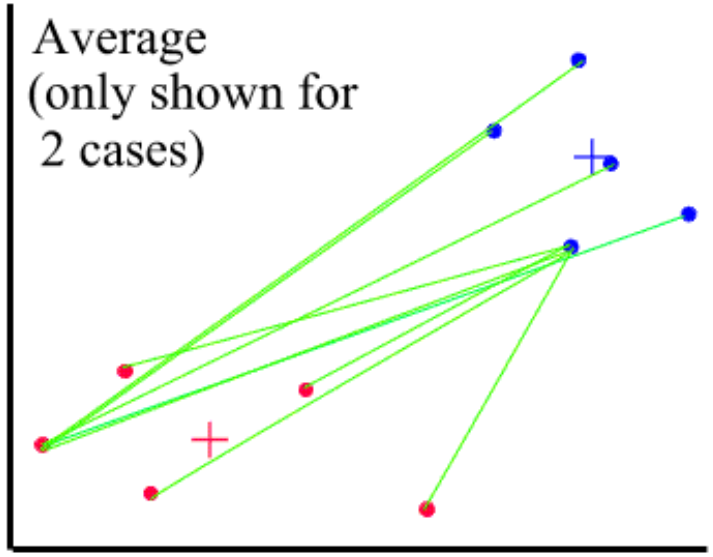
Defines the distance between the two clusters as the maximum distance over all possible pairs. It will merge the two clusters with the minimum resulting diameter.

It's more likely to get spherical clusters with consistent diameter.



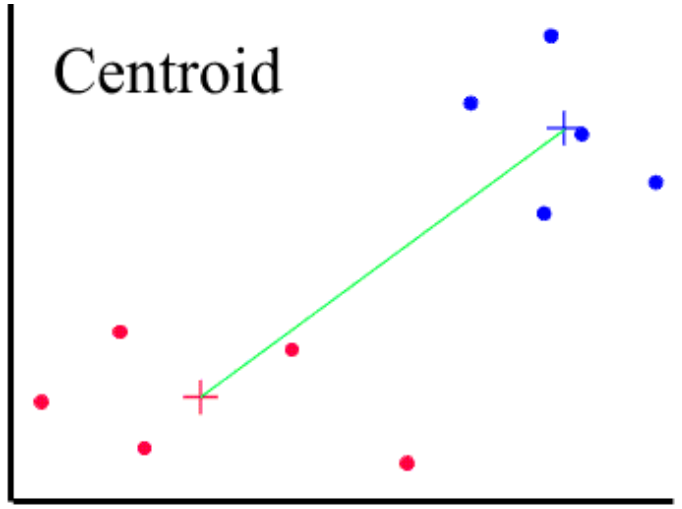
Average Linkage / Between Groups Linkage

The distance between the two clusters is an average of all pairwise distances across the two clusters. The smallest average distance between all group pairs is computed and the two groups that are closest are combined. This approach is less affected by the outliers.



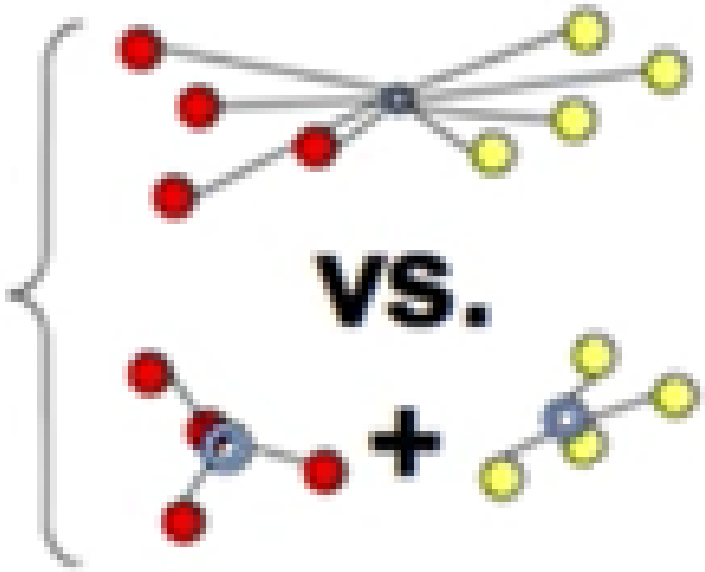
Centroid Clustering

The centroids (means) of each cluster are calculated. It will merge the two clusters with the minimum distance between the centroids.



Ward's Method

It's one of the best metrics and is widely used. It pretends to merge the two clusters based on the aggregate deviation of the result. If we want to merge two clusters, we estimate the centroid of the resulting cluster and then look for the sum of the squared deviations of all the points from the new centroid. For different merging pairs of clusters we will get different deviations. And we pick the merge that results in the smallest deviation.



k-means cluster analysis

Is recommended for a moderately sized data set if the number of clusters is known beforehand. The algorithm iteratively estimates the cluster means and assigns each case to the cluster for which its distance to the cluster mean is the smallest.

It is used when the researcher already has hypotheses concerning the number of clusters in the analyzed cases. Very often hierarchical cluster analysis is performed first on a small sample to determine the number of clusters and then the k-meanы cluster analysis is used to classify all the cases.

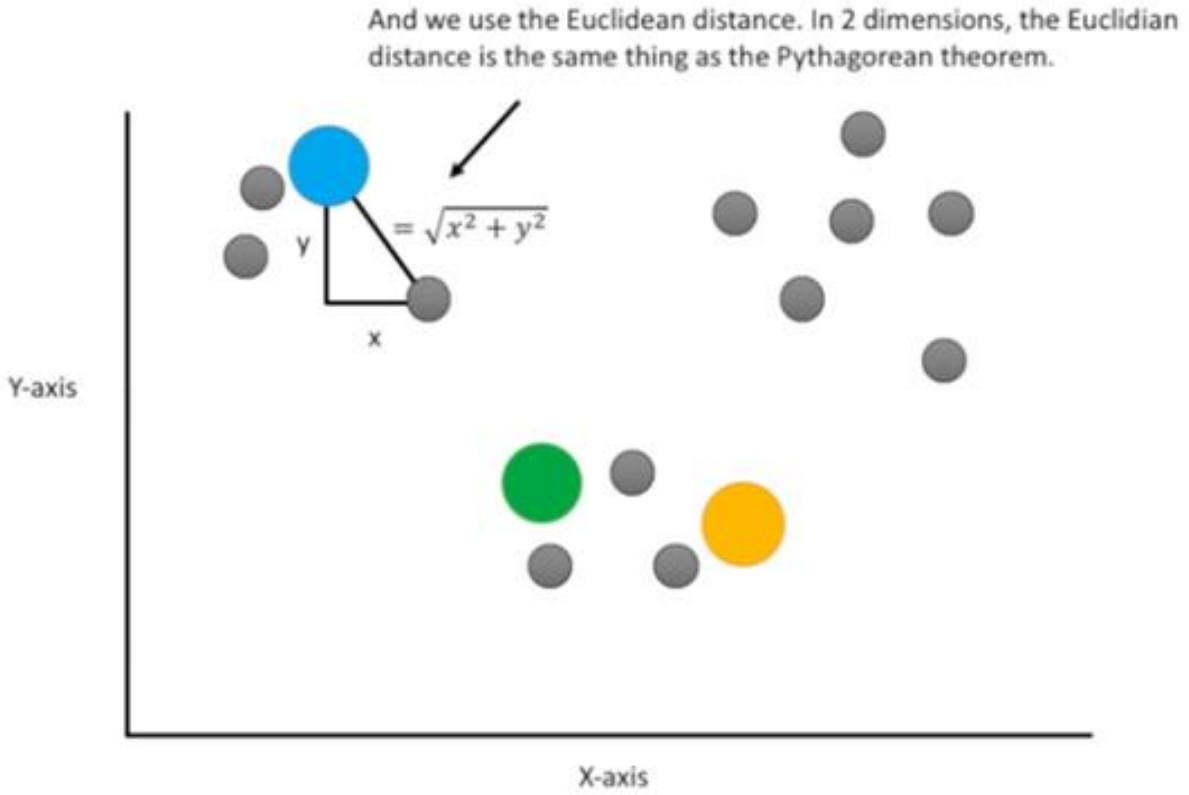
k-means cluster analysis: step 1

Select the number of clusters you want to identify in the data. Your decision could be based on:

- your research task,
- results of other studies on a similar topic,
- preliminary hierarchical cluster analysis based on small subsamples taken from the sample.

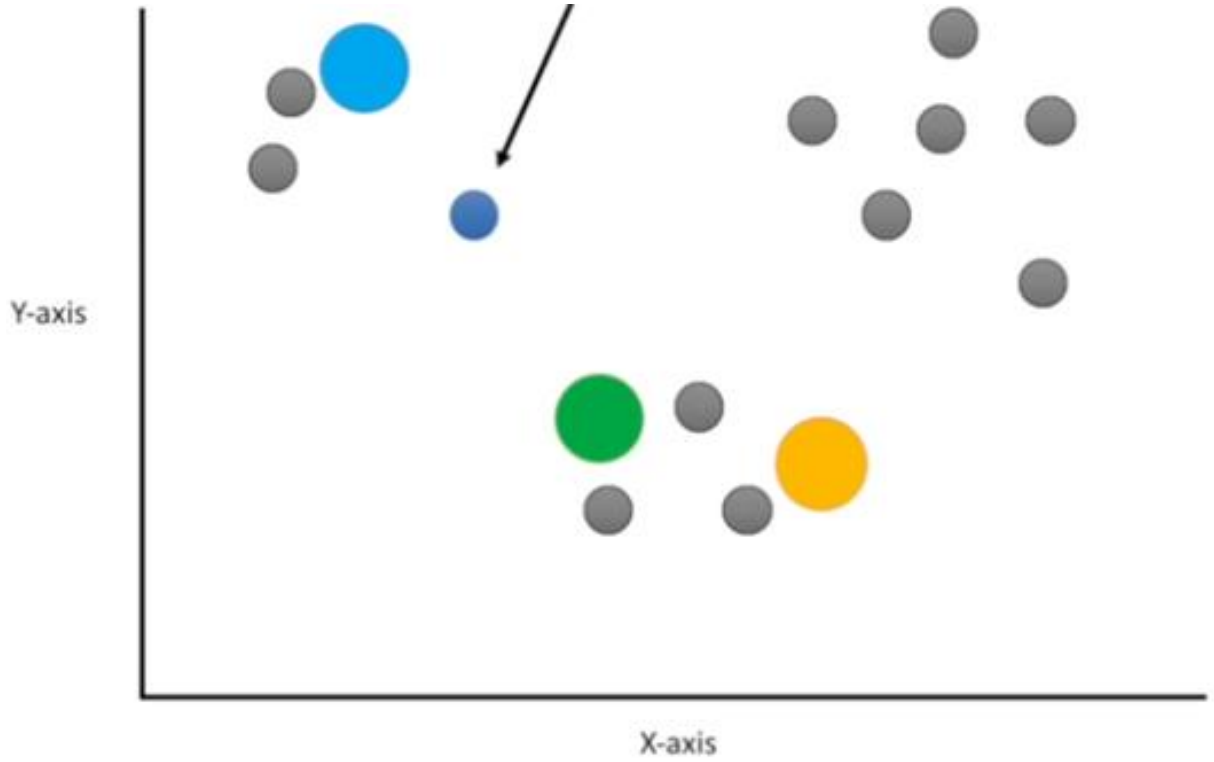
k-means cluster analysis: step 2

Pick the random points according to the number of clusters that was selected at the previous step.



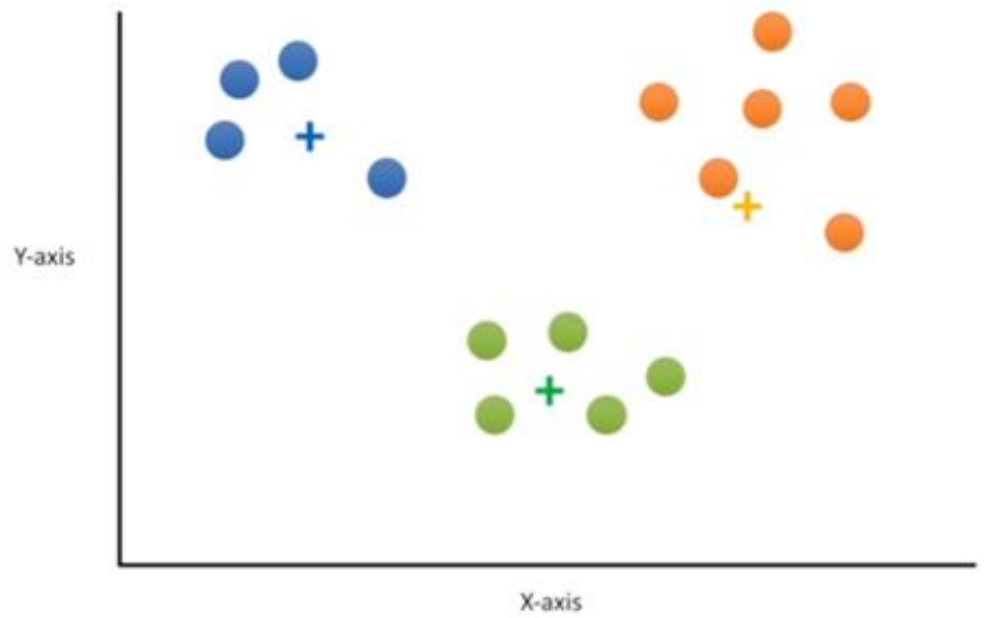
k-means cluster analysis: step 3

Assign the points to the nearest clusters.



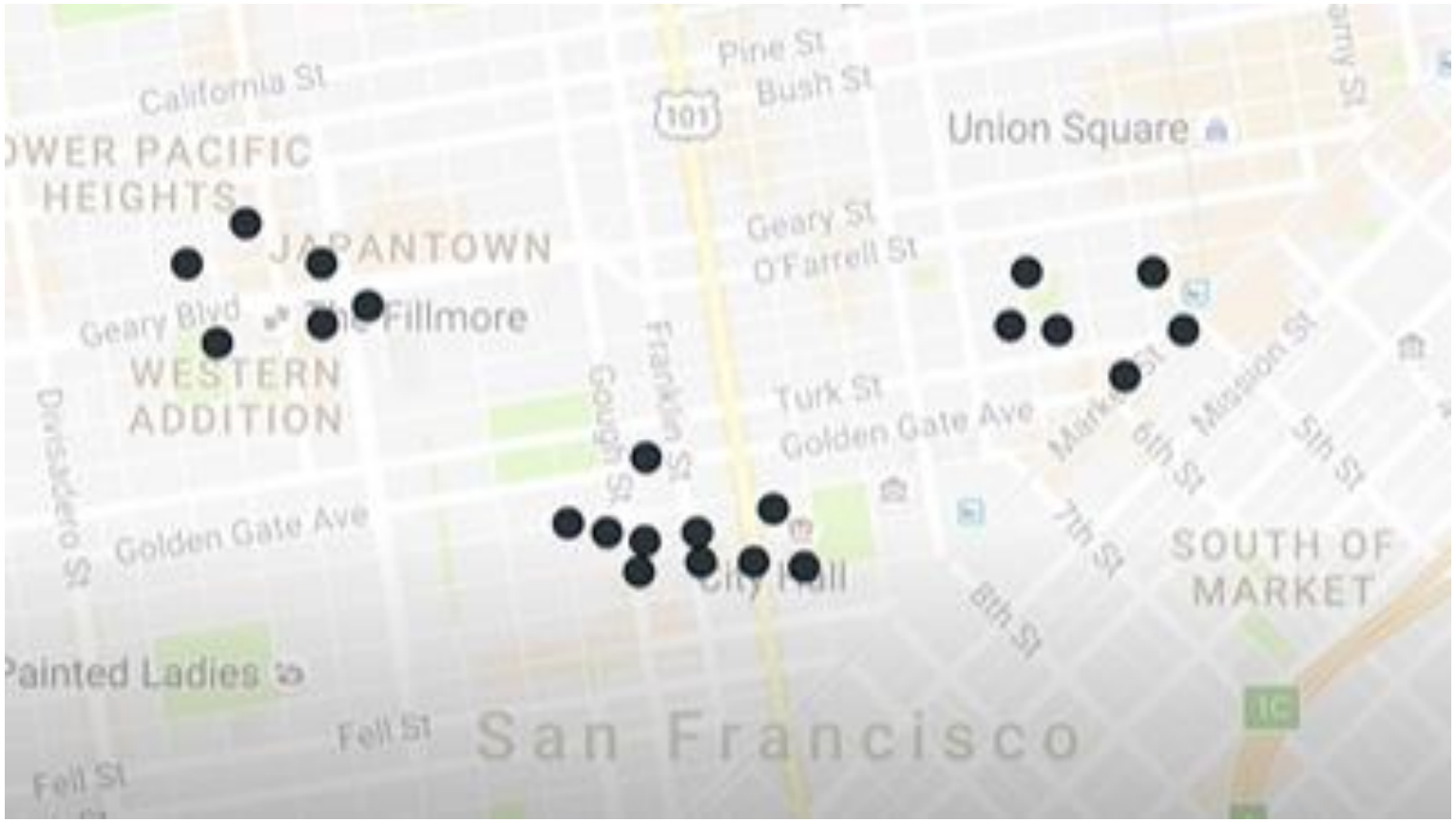
k-means cluster analysis: step 4

Calculate the center of each cluster and recluster. Reclustering should be repeated until no improvements are possible.



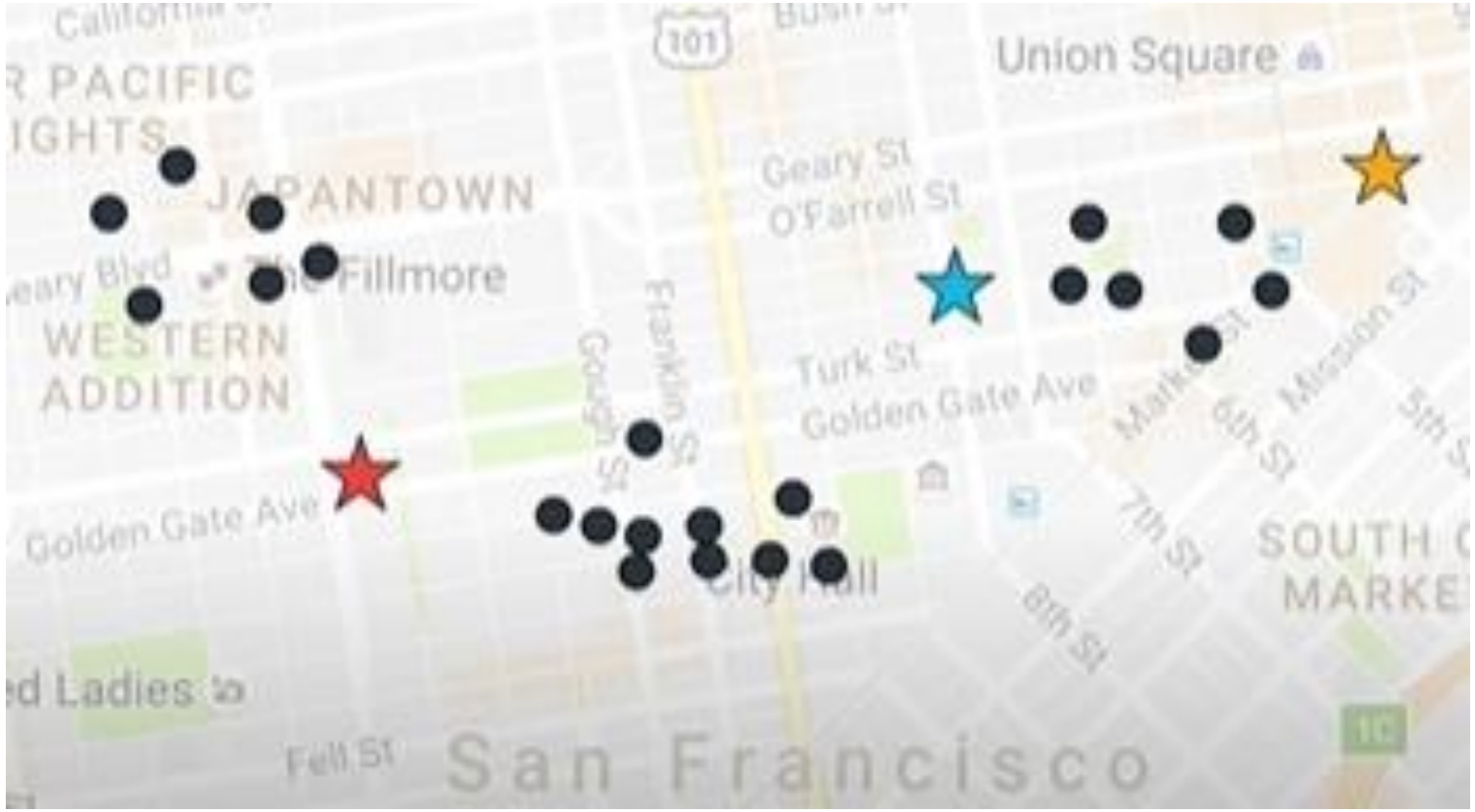
k-means clustering: example

We want to select the best positions for three cafes.



k-means clustering: example

First, we locate three random points.



k-means clustering: example

Second, we connect the points to the closest center.

k-means clustering: example

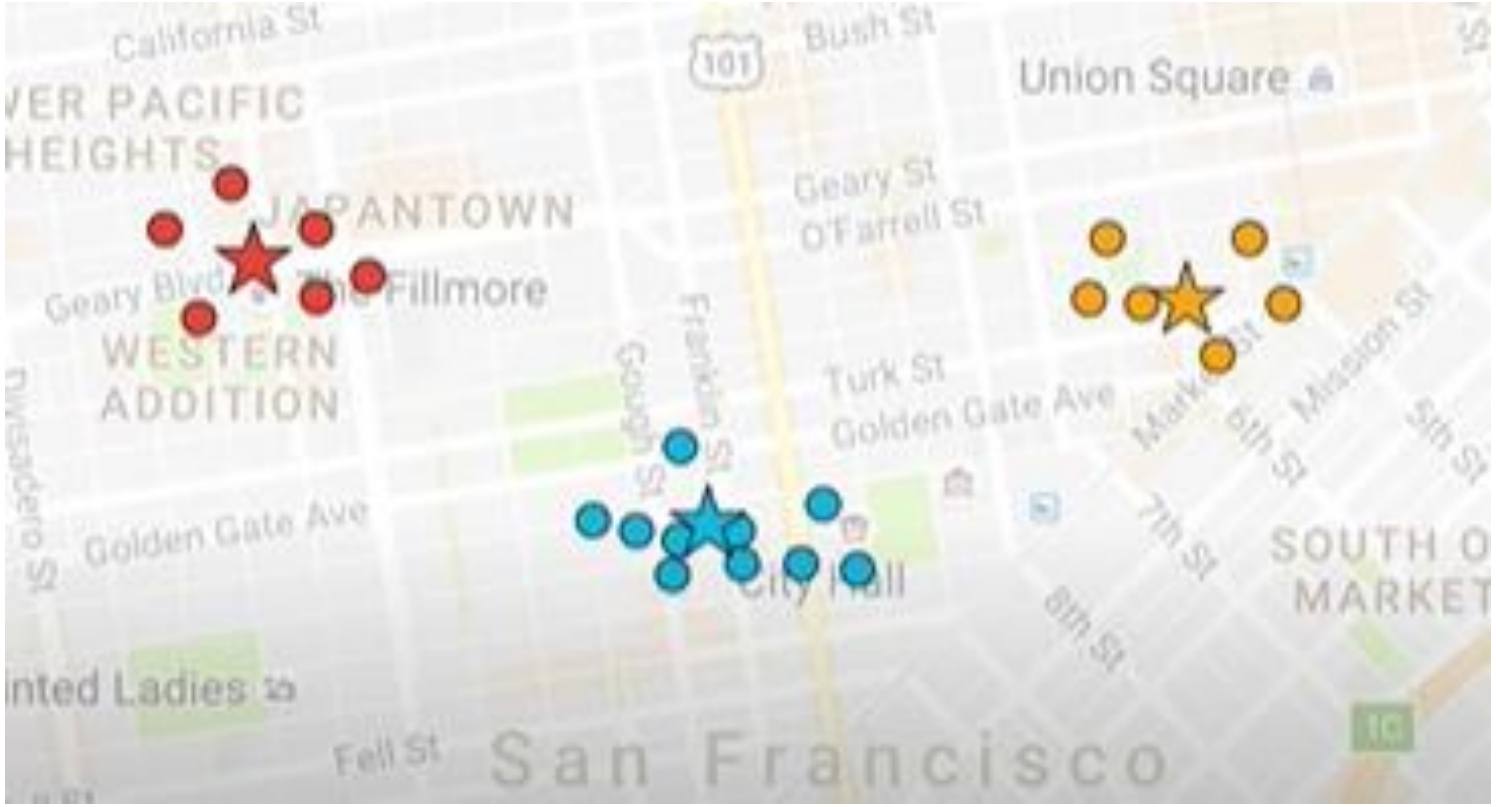
Third, we put the central points to the center of the houses that they are serving.

k-means clustering: example

Forth, we recalculate the results to minimize the distance between each building and cafes that we have. And then we again recalculate the centers.

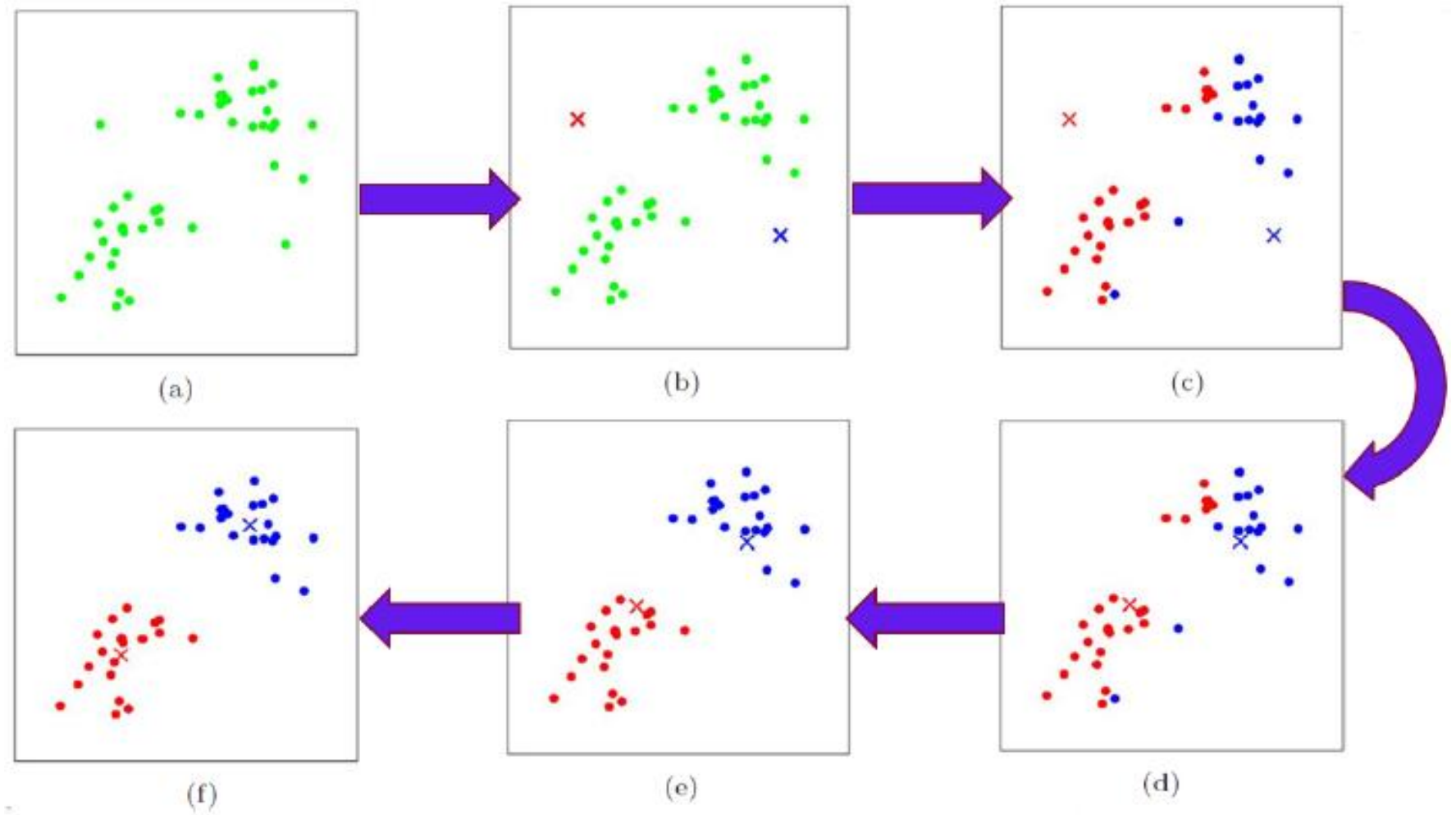
k-means clustering: example

Final result.

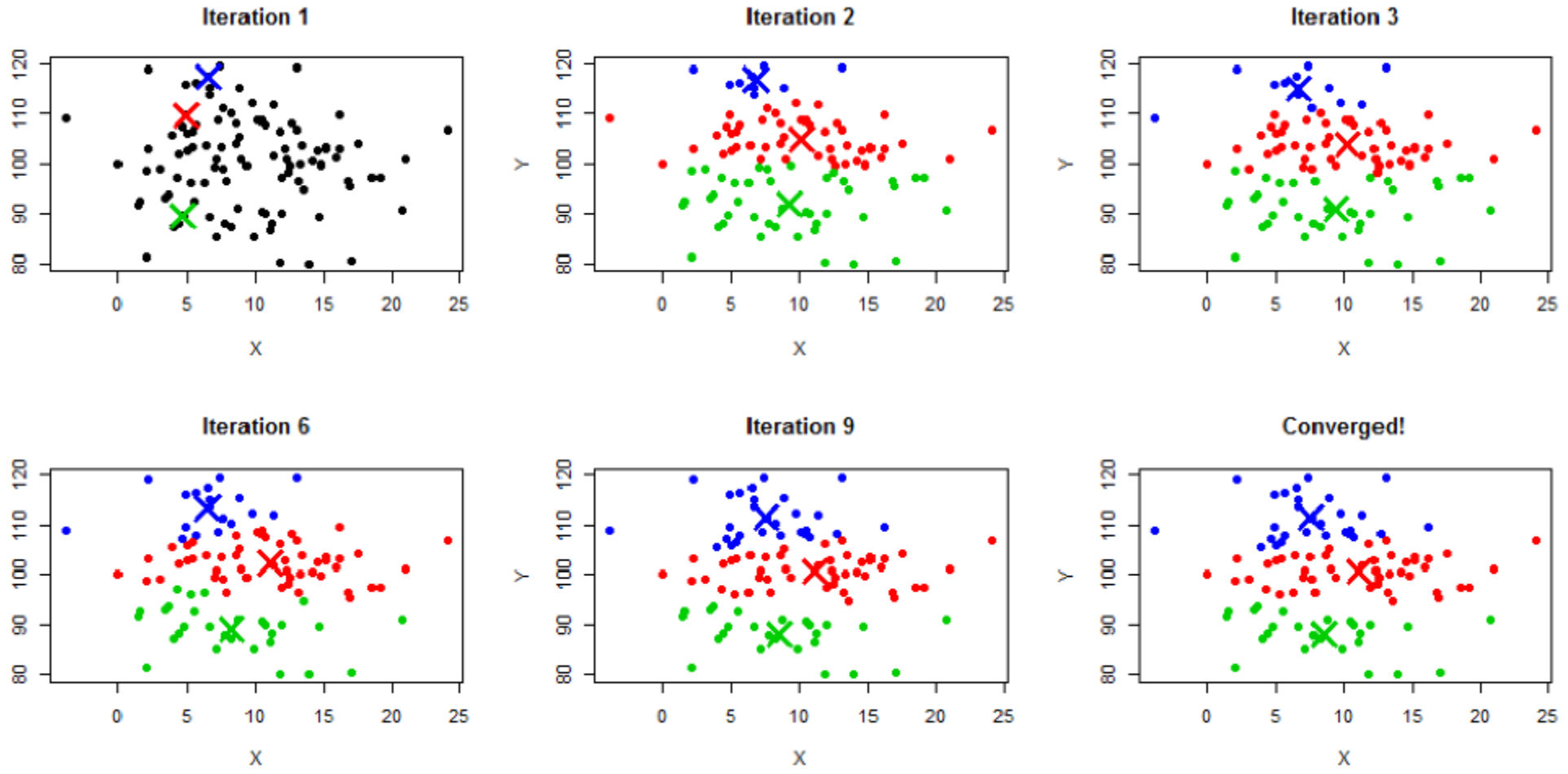


Source: https://www.youtube.com/watch?v=QXOkPvFM6NU

k-means clustering (2 clusters)



k-means clustering (3 clusters)



k-medians

K-medians is a clustering algorithm similar to K-means, but it uses the median as the cluster center. This makes K-medians more robust to outliers. It minimizes Manhattan distance.

K-medians relies on the median, which only requires knowing the order of values, not their exact distance. That makes it well-suited for ordinal data, compared to K-means, which uses the mean and assumes numeric scale.

k-modes

K-modes is designed specifically for categorical data. It uses modes (most frequent category) as cluster center. To calculate the distance it uses matching dissimilarity (count of mismatches between categories).

k-prototypes

K-Prototypes is a clustering algorithm for datasets with both numerical and categorical features. It combines K-Means (using means and Euclidean distance for numeric data) and K-Modes (using modes and mismatch count for categorical data). Each cluster is represented by a prototype made of numeric averages and categorical modes, and distance is a weighted mix of numeric and categorical dissimilarity. This makes it useful for real-world mixed data.

K-Family of Clustering Algorithms

Algorithm	Data Type	Cluster Center	Distance Metric	Robust to Outliers?	Example Use
K-Means	Numerical	Mean	Euclidean (L2)	❌ No	Income, Age
K-Medians	Numerical/Ordinal	Median	Manhattan (L1)	✅ Yes	Satisfaction rating
K-Modes	Categorical	Mode	Mismatch count	N/A	Job role, Education
K-Prototypes	Mixed (Num + Cat)	Mean + Mode	L2 + Mismatch combo	Partial	Age + Job role + Dept

k-means++

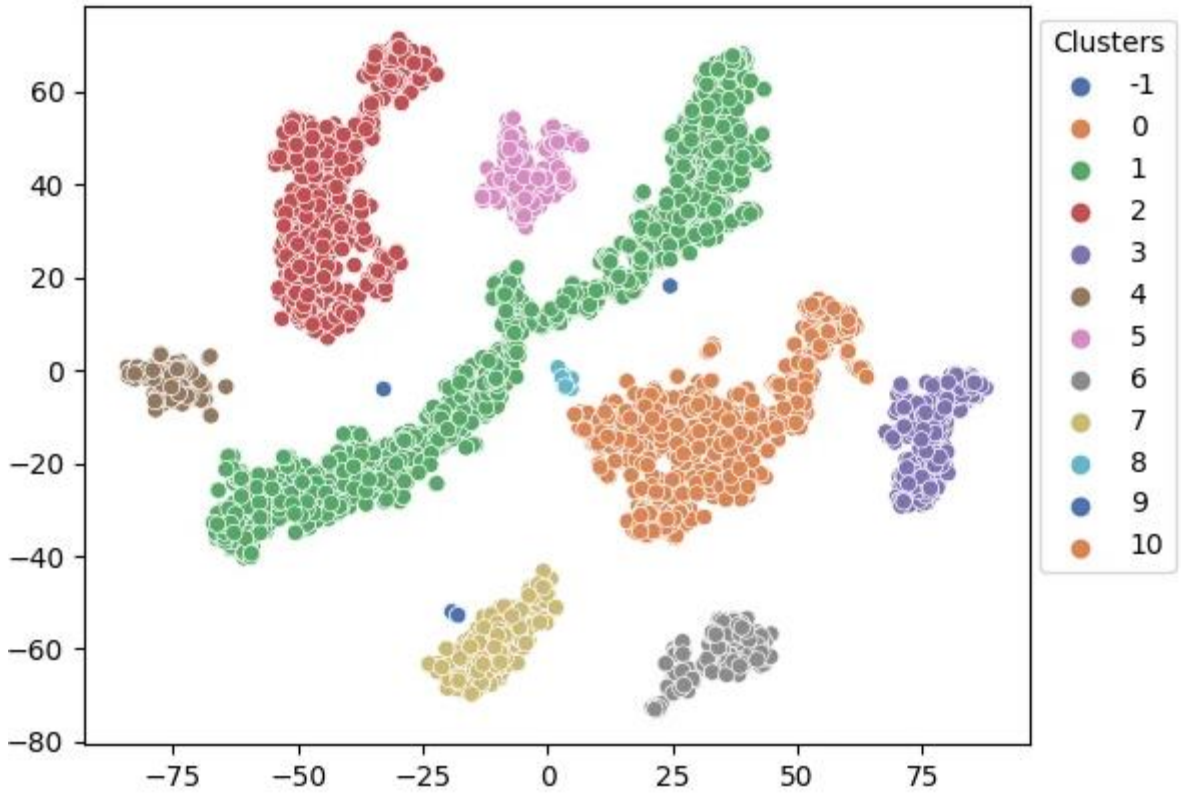
K-means initial cluster centers are chosen randomly. If the starting points are poorly chosen algorithm can converge to a bad local minimum, clusters may be imbalanced, and results can vary a lot between runs.

K-Means++ improves initialization by spreading out the initial centroids before running the regular K-Means algorithm. This increases the chance of finding good clusters, reduces randomness, and speeds up convergence.

DBSCAN

DBSCAN (Density-Based Spatial Clustering of Applications with Noise) is a powerful tool to identify nested clusters and works better for data with complex structure. It is robust to outliers and classifies them as separate cluster.

DBSCAN uses density of points to identify clusters: regions with high density of points are considered as clusters, while low-density are assumed to be outliers.



DBSCAN

DBSCAN groups together closely-packed points. The most stark difference between DBSCAN and other clustering algorithms is that not every point is part of a cluster; these points are considered *noise*. Other points are classified as either a core point or border point.

There are two inputs to DBSCAN:

- *epsilon* – the maximum distance between 2 points for them to be considered neighbors of one another (e.g.: 0.01, 0.5, 2 etc.);
- *min_samples* – the minimum number of neighbors a point needs to have to be considered a core point (e.g.: 3, 5, 10 etc.).

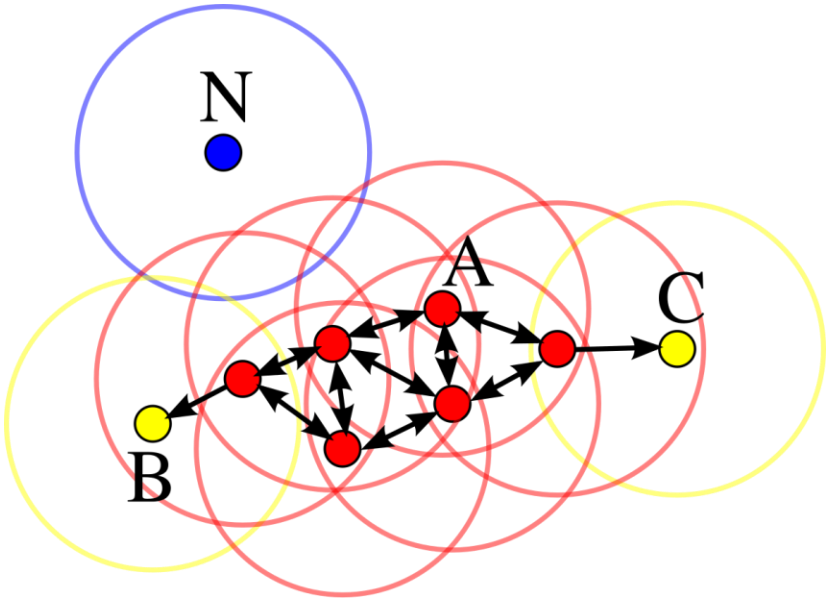
DBSCAN

DBSCAN clustering algorithm works by dividing the data points into three categories:
core points, **border points**, and **noise points**.

The algorithm starts by selecting a random data point and checking its neighboring points.

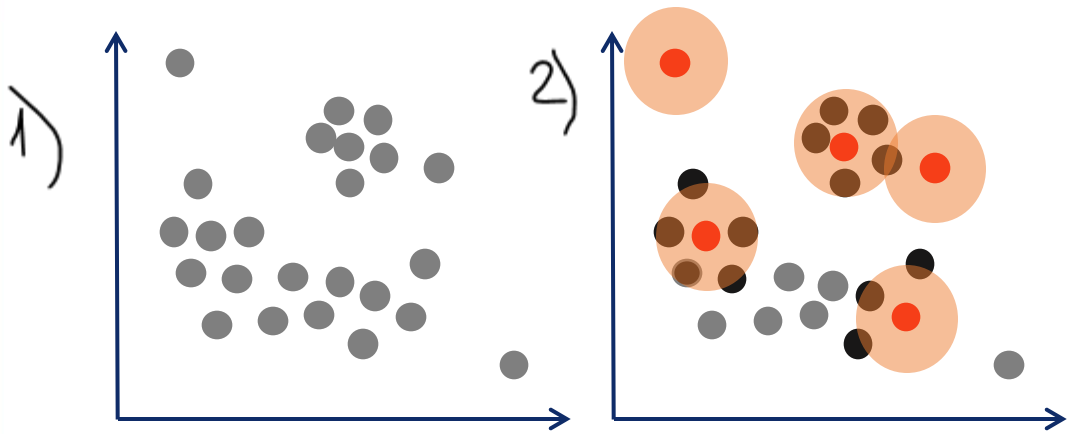
- If the number of neighboring points is **greater than or equal to** a specified threshold value, the point is considered a **core point**.
- If the number of neighboring points is **less than the threshold value**, the point is considered a **border point**.
- If a point **does not have any neighboring points**, it is considered a **noise point**.

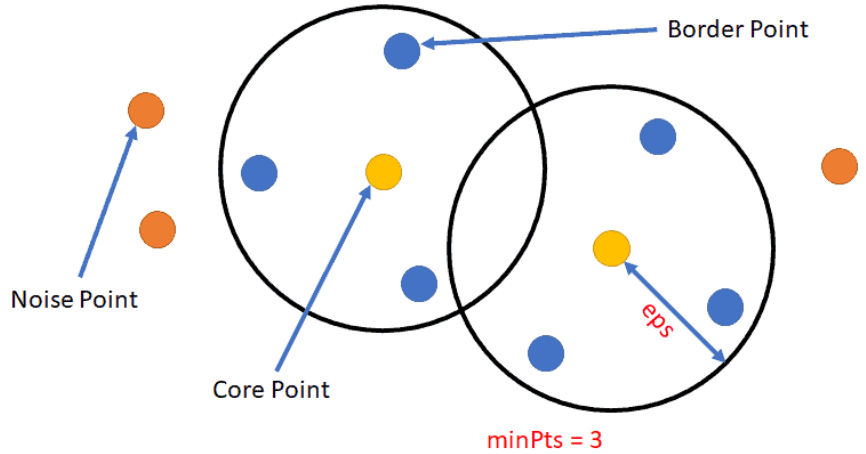
Once all the points are classified, the algorithm starts forming clusters. The algorithm creates a cluster by connecting all the core points and their border points. If two core points are close enough to each other, they are considered part of the same cluster. The algorithm continues forming clusters until all the core points are assigned to a cluster.



DBSCAN algorithm

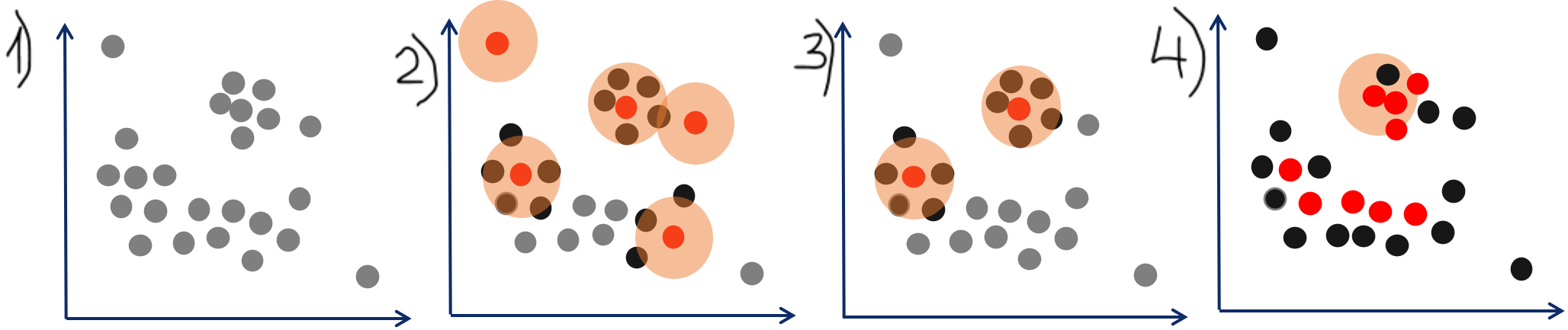
1) Create a space of points
2) Count the number of points close to each point (based on radius – epsilon). Define a Core point – a point that has at least N neighbors (N is a parameter) within a radius of epsilon.





DBSCAN algorithm

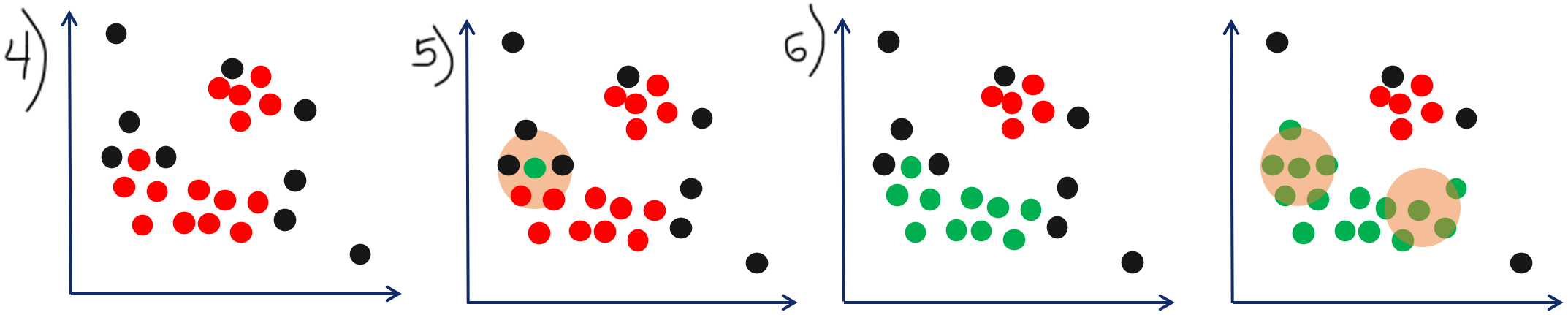
3) Find all core points that have at least N neighbors (red in the example)
4) The remaining points are considered non-core points (black points)



Good reference to read in Russian: https://habr.com/ru/articles/322034/

DBSCAN algorithm

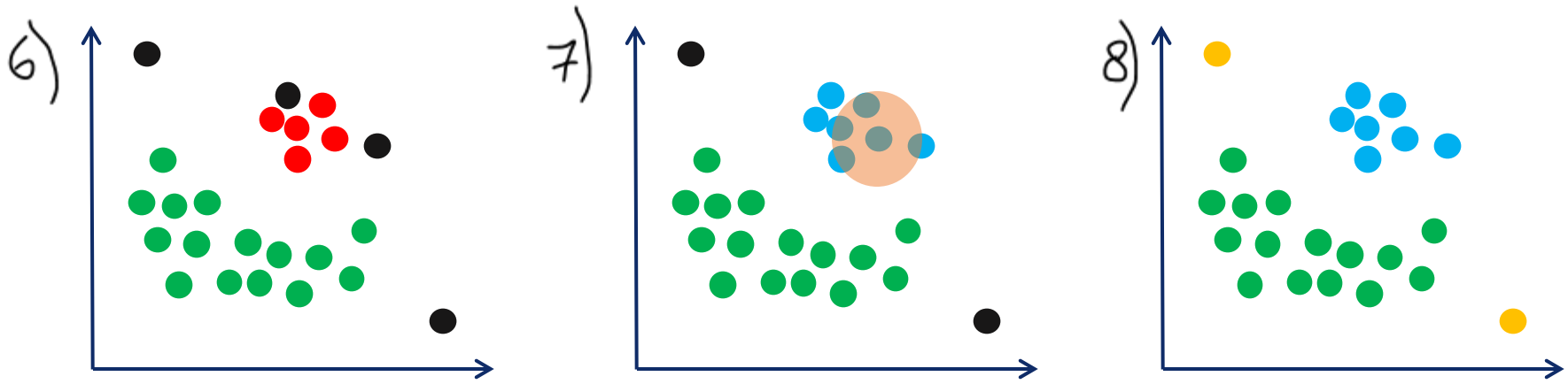
5) Randomly pick a core point and assign it to the first cluster
6) The core points near the selected point are added to the first cluster. Repeat until the cluster can no longer be extended. When no more core points are left to add to the first cluster, add non-core points that are close to the core points of the first cluster.



DBSCAN clustering algorithm

7) The other points that are close to each other, but are separated from the first cluster, are used to form the second cluster (in the same way).

8) When all core points are assigned to clusters, the clustering is finished. The points that do not belong to any cluster are called outliers.



k-means benefits and limitations

Benefits

- Simple and easy to understand;
- Fast and scalable, making it suitable for large datasets.

Limitations

- Requires the number of clusters to be specified in advance;
- Can be sensitive to the initial placement of cluster centroids;
- May not work well with data that has complex shapes or overlapping clusters;
- Can be affected by outliers or noise in the data.

Hierarchical Clustering benefits and limitations

Benefits

- Produces a dendrogram that shows the relationships between data points and clusters;
- Does not require the number of clusters to be specified in advance.

Limitations

- Can be computationally expensive and slow for large dataset;
- May not work well with data that has complex shapes or overlapping clusters;
- Can be sensitive to the choice of similarity metric and linkage method.

DBSCAN benefits and limitations

Benefits

- Can identify clusters of varying shapes and sizes;
- Can handle noise and outliers in the data;
- Does not require the number of clusters to be specified in advance.

Limitations

- Requires the specification of two parameters: epsilon and the minimum number of points required to form a cluster;
- Can be sensitive to the choice of parameters and the distance metric used;
- May not work well with data that has varying densities or complex shapes;
- Can be computationally expensive for large datasets.

Validating the results

To validate the results, we can use ANOVA test to check that the mean values of the variables are significantly different by clusters.

We can assess the quality of the clustering by calculating the variation of variables' values within each cluster.

Cluster profiles

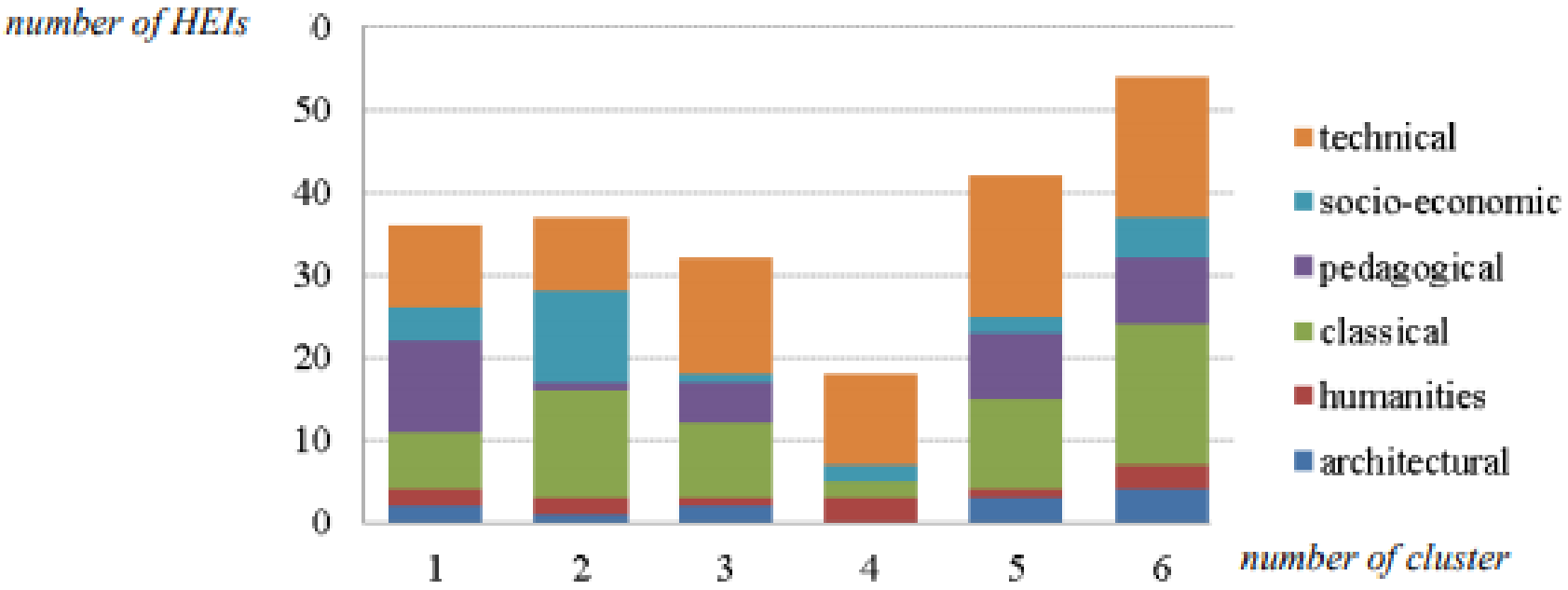
Table 1. Cluster profiles and object distribution

Cluster number	USE	Youngsters	UPF	Size	R&D	Number of objects
1	59.81	0.24	82.75	5905.31	89.82	36
2	66.32	0.15	90.05	13495.43	86.92	37
3	57.42	0.06	95.82	5825.06	71.61	32
4	70.77	0.16	205.40	7947.39	70.72	18
5	59.53	0.21	85.15	7177.98	50.80	42
6	65.00	0.15	101.54	9005.85	13.41	54

Source: [*Abankina et al.*, 2013b]

Comparing clusters

Figure 3. Distribution of various types of universities in the proposed typology



Source: [*Abankina et al., 2013b*] *and figures for* architectural and humanities HEIs are calculated by the authors.

Optimal number of clusters: CH Index

Calinski and Harabasz Index (CH Index) is often used for choosing the optimal number of clusters. This criterion combines the within and between cluster variance to evaluate the quality of segmentation. The formula of ***CH Index*** defined as:

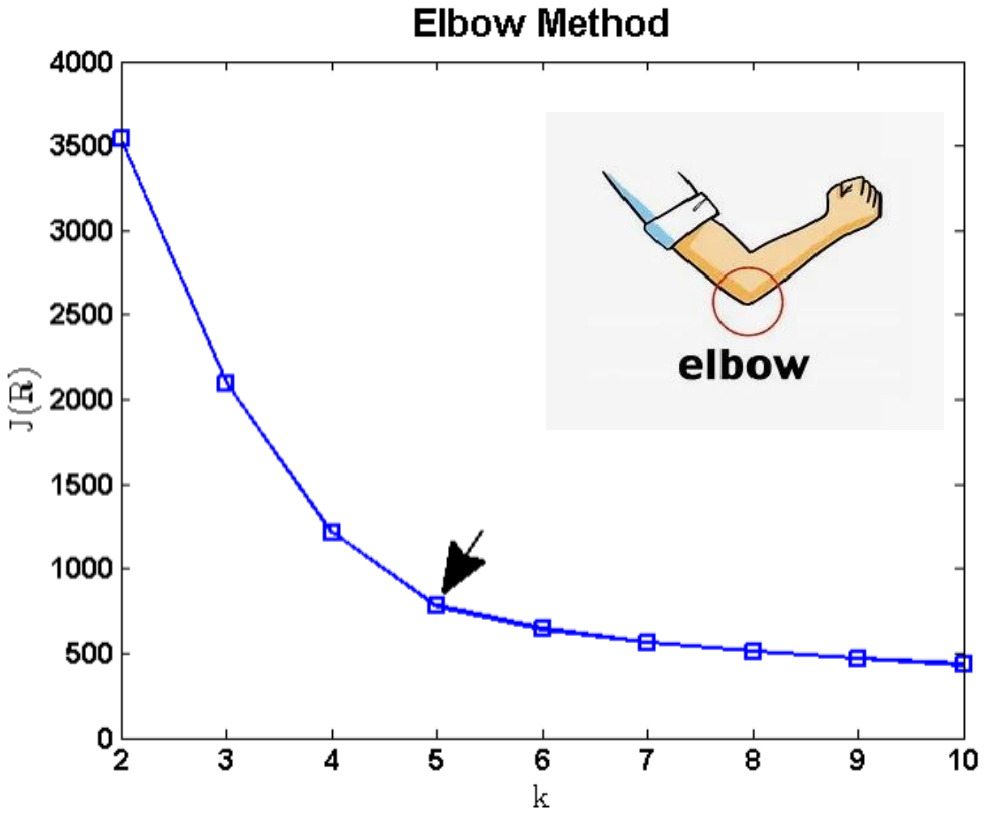
$$CH(K) = \frac{B(K)/(K-1)}{W(K)/(n-K)}$$

where ***W*** is within-cluster variation, ***B*** is between-cluster variation, K is number of clusters, n is sample size. To obtain a small within-cluster variation and large between-cluster variation simultaneously, we would pick the value of K with the largest CH score as the stopping point.

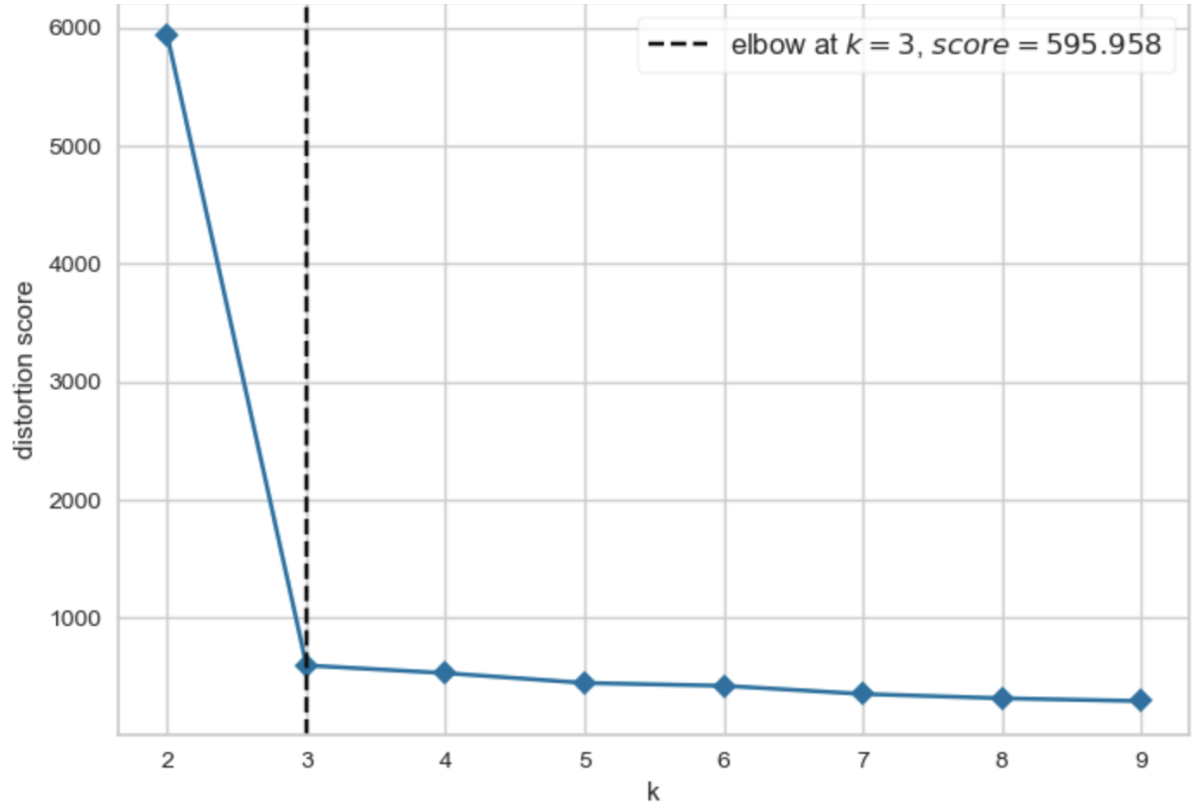
Optimal number of clusters: elbow method with Inertia

We can show on the graph the value of **Inertia** for different number of clusters. **Inertia** is calculated by measuring the distance between each data point and its centroid, squaring this distance, and summing these squares across one cluster. A good model is one with low inertia AND a low number of clusters. However, this is a tradeoff because as K increases, inertia decreases.

Most frequently this approach is used with k-means.



Optimal number of clusters: elbow method with Distortion Score

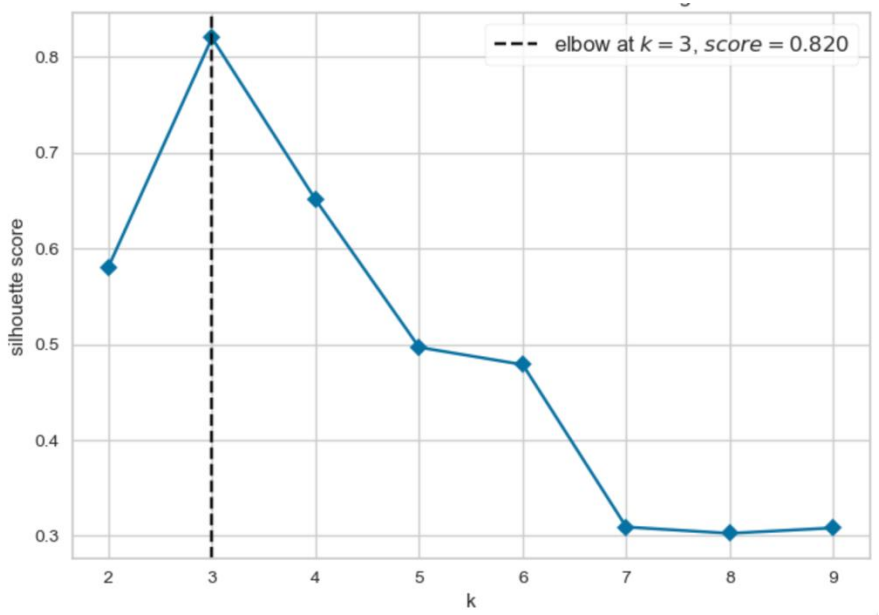


Distortion Score is the average of the Euclidean squared distance from the centroid of the respective clusters.

Optimal number of clusters: Silhouette Score

The **Silhouette Score** measures how similar the data points inside the cluster are compared to how different the clusters are from each other. The silhouette score calculation involves two core components: cohesion and separation. Cohesion measures the similarity of the points within the cluster. Separation shows the degree to which the clusters don't overlap.

Silhouette values lie in the [-1, 1] interval, with -1 indicating a misclassified point, 1 indicating that the point is closely tied with its cluster and poorly matched with the neighboring clusters, and a score of 0 means that there is no clear separation between the clusters and they might overlap. Generally speaking, scores of 0.7 and higher are considered acceptable.



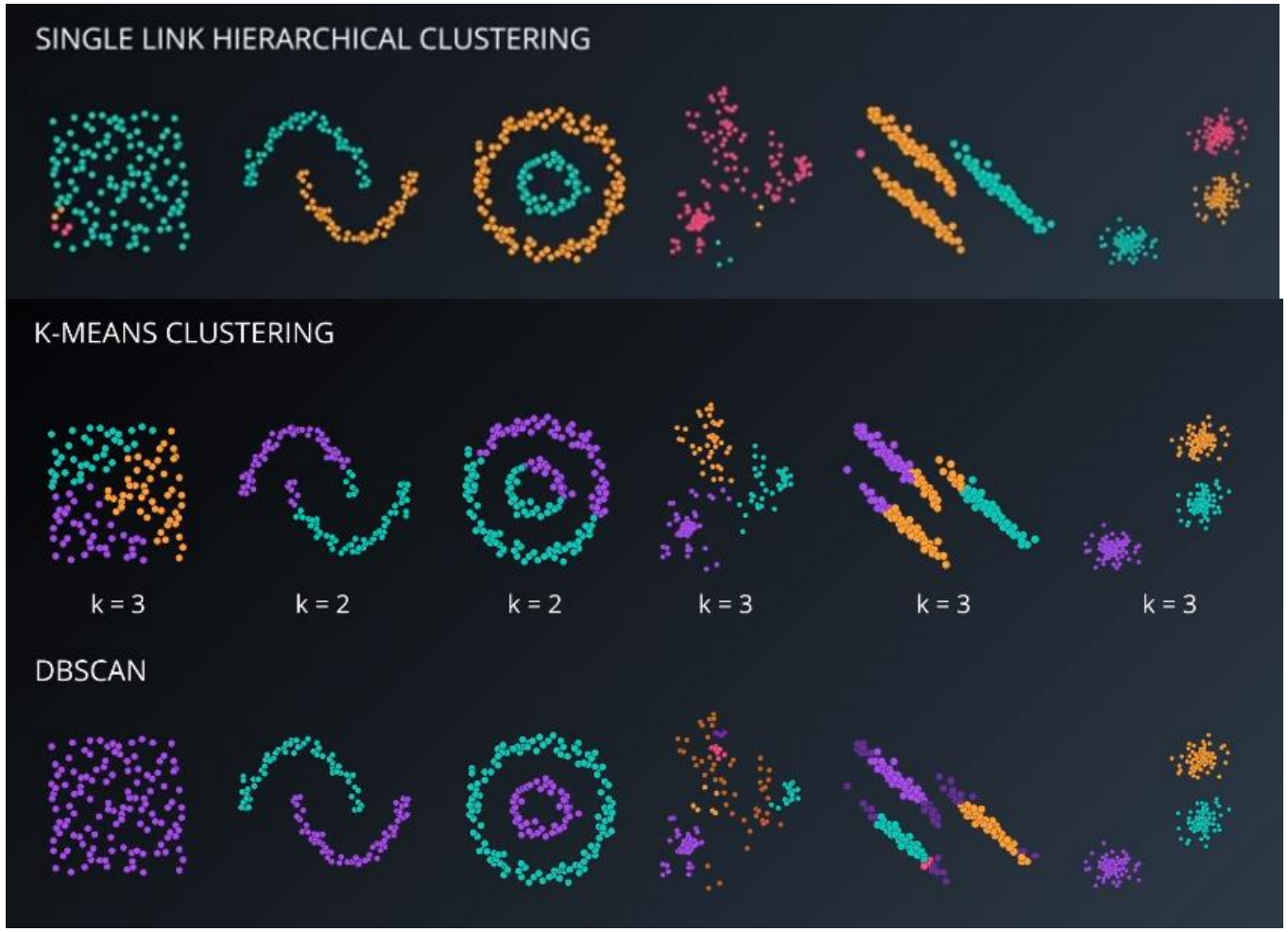
Optimal number of clusters

Metric	Mathematical Basis	Primary Use
Inertia	Sum of squared distances from points to centroids	Measures compactness for optimal k with Elbow Method
Distortion	Average distance from points to centroid	Elbow Method for optimal cluster selection
Silhouette	Ratio of cohesion and separation	Cross-model comparison in clustering

Visualizing clustering algorithms

- Visualizing K-Means Clustering (naftaliharris.com)
- Visualizing DBSCAN Clustering (naftaliharris.com)

Comparing 3 algorithms



Thank you for your attention!